Ground Texture Based Localization

Dissertation
zur Erlangung des Doktorgrades
der Naturwissenschaften

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

von
Jan Fabian Schmid
aus Flensburg

Hildesheim 2023
D 30

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

Dekan:	Prof. Dr. Martin Möller
1. Gutachter:	Prof. Dr. Rudolf Mester
2. Gutachter:	Prof. Dr. Darius Burschka
Datum der Disputation:	14.12.2023

Abstract

This dissertation is concerned with the task of map-based self-localization, using images of the ground recorded with a downward-facing camera. In this context, map-based (self-)localization is the task of determining the position and orientation of a query image that is to be localized. The map used for this purpose consists of a set of reference images with known positions and orientations in a common coordinate system. For localization, the considered methods determine correspondences between features of the query image and those of the reference images.

In comparison with localization approaches that use images of the surrounding environment, we expect that using images of the ground has the advantage that, unlike the surrounding, the visual appearance of the ground is often long-term stable. Also, by using active lighting of the ground, localization becomes independent of external lighting conditions.

This dissertation includes content of several published contributions, which present research on the development and testing of methods for feature-based localization of ground images. Our first contribution examines methods for the extraction of image features that have not been designed to be used on ground images. This survey shows that, with appropriate parametrization, several of these methods are well suited for the task.

Based on this insight, we develop and examine methods for various subtasks of map-based localization in the following contributions. We examine global localization, where all reference images have to be considered, as well as local localization, where an approximation of the query image position is already known, which allows for disregarding reference images with a large distance to this position.

In our second contribution, we present the first systematic comparison of state-of-the-art methods for ground texture based localization. Furthermore, we present a method, which is characterized by its usage of our novel feature matching technique. This technique is called *identity matching*, as it matches only those features with identical descriptors, in contrast to the state-of-the-art

that also matches features with similar descriptors. We show that our method is well suited for global and local localization, as it has favorable scaling with the number of reference images considered during the localization process. In another contribution, we develop a variant of our localization method that is significantly faster to compute, as it applies a sampling approach to determine the image positions at which local features are extracted, instead of using classical feature detectors.

Two further contributions are concerned with global localization. The first one introduces a prediction model for the global localization performance, based on an evaluation of the local localization performance. This allows us to quickly evaluate any considered parameter settings of global localization methods. The second contribution introduces a learning-based method that computes compact descriptors of ground images. This descriptor can be used to retrieve the overlapping reference images of a query image from a large set of reference images with little computational effort.

The most recent contribution included in this dissertation presents a new ground image database, which was recorded with a dedicated platform using a downward-facing camera. In addition to the data, we also explain our guidelines for the construction of the platform. In comparison with existing databases, our database contains more images and presents a larger variety of ground textures. Furthermore, this database enables us to perform the first systematic evaluation of how localization performance is affected by the time interval between the point in time at which the reference images are recorded and the point in time at which the query image is recorded. We find out that for outdoor areas all ground texture based localization methods have reliability issues, if the time interval between the recording of the query and reference images is large, and also if there are different weather conditions. These findings point to remaining challenges in ground texture base localization that should be addressed in future work.

Kurze Zusammenfassung

Diese Dissertation beschäftigt sich mit der Aufgabe der kartenbasierten Eigenlokalisierung anhand von Bodenbildern, die mit einer nach unten gerichteten Kamera aufgenommen werden. Als kartenbasierte (Eigen-)Lokalisierung bezeichnen wir die Bestimmung der Position und Orientierung eines zu lokalisierenden Bildes in einer Karte. Die dabei verwendete Karte wird aus Referenzbildern zusammengesetzt, deren Position und Orientierung zuvor bestimmt wurde. Für die Lokalisierungsaufgabe werden Methoden betrachtet, die Korrespondenzen zwischen Bildmerkmalen aus diesen Referenzbildern und einem zu lokalisierenden Bild identifizieren. Wir erwarten, dass ein Vorteil dieses Ansatzes gegenüber der Lokalisierung anhand von Bildern der Umgebung darin bestehen kann, dass das visuelle Erscheinungsbild des Bodens langfristig stabiler ist als das der Umgebung. Durch den Einsatz einer aktiven Beleuchtung des Bodens wird die Lokalisierung zudem unabhängig vom Umgebungslicht.

In den in dieser Dissertation präsentierten Beiträgen werden neue Verfahren für die bildmerkmalsbasierte Lokalisierung von Bodenbildern entwickelt und getestet. Unser erster Beitrag untersucht Bildmerkmalsextraktionsmethoden, die ursprünglich nicht für Bodenbilder entwickelt wurden, auf ihre Eignung für die bodentexturbasierte Lokalisierung. Dabei zeigt sich, dass bei passender Parametrisierung einige dieser Methoden gut für die Aufgabe geeignet sind.

Ausgehend von dieser Erkenntnis entwickeln und untersuchen wir in den darauf folgenden Beiträgen Verfahren für verschiedene Teilaufgaben der kartenbasierten Lokalisierung. Wir betrachten sowohl den Fall der globalen Lokalisierung, bei der alle Referenzbilder in Betracht gezogen werden müssen, sowie die lokale Lokalisierung bei der die aktuelle Position bereits ungefähr bekannt ist, sodass weit von dieser Position entfernte Referenzbilder bei der Lokalisierung nicht berücksichtigt werden müssen. In unserem zweiten Beitrag präsentieren wir einen systematischen Vergleich aktueller Methoden zur bodentexturbasierten Lokalisierung, und untersuchen dabei detailliert die Lokalisierungsleistung der Methoden für die globale und lokale Lokalisierung. Zudem stellen wir eine neue Lokalisierungsmethode vor, die sich insbesondere durch eine neue von uns eingeführte Technik zur Korrespondenzfindung zwischen den

Bildmerkmalen auszeichnet. Dabei handelt es sich um den Identitätsabgleich, bei dem ausschließlich Merkmale mit identischen Merkmalsbeschreibungen als mögliche Korrespondenzen berücksichtigt werden. Dies steht im Kontrast zu üblichen Verfahren der Korrespondenzfindung, die auch ähnliche Deskriptoren berücksichtigen. Wir zeigen, dass unsere Methode gut für beide Lokalisierungsmodi geeignet ist, da sie vorteilhaft mit der Anzahl der berücksichtigten Referenzbilder skaliert. In einem weiteren Beitrag entwickeln wir eine Variante dieser Lokalisierungsmethode, die durch den Einsatz eines Stichprobenverfahrens anstatt eines klassischen Merkmalsdetektionsverfahrens deutlich weniger Rechenaufwand verursacht.

In zwei weiteren Beiträgen stellen wir Verfahren für die globale Lokalisierung vor. Zum Einen entwickeln wir eine modellbasierte Vorhersage, welche auf Grundlage einer Evaluation der lokalen Lokalisierungsleistung einer Methode, dessen Erfolgsrate bei der globalen Lokalisierung prädiziert. Diese Vorhersage nutzen wir zur schnelleren Auswertung möglicher Parametrisierungen von globalen Lokalisierungsmethoden. Zum Zweiten entwickeln wir ein neues Verfahren des maschinellen Lernens für die Beschreibung von Bodenbildern anhand eines kompakten Deskriptors. Dieser Deskriptor kann dafür genutzt werden, die überlappenden Referenzbilder eines zu lokalisierenden Bildes mit geringem Rechenaufwand aus einer großen Menge Referenzbilder herauszusuchen.

Der chronologisch letzte Beitrag, der in diese Dissertation aufgenommen wurde, präsentiert eine neue Datenbasis von Bodenbildern. Diese wurde mit einer eigens dafür aufgebauten Aufnahmeplattform mit nach-unten-gerichteter Kamera aufgenommen. Die Leitlinien für den Aufbau dieser Plattform präsentieren wir ebenfalls. Unsere Datenbasis enthält im Vergleich zu vorhandenen Datenbasen deutlich mehr Bilder und bietet eine größere Anzahl verschiedener Bodentexturen. Außerdem können wir mit dieser Datenbasis erstmals systematisch untersuchen, wie die Lokalisierungsleistung von der Zeitspanne zwischen dem Zeitpunkt der Kartierung des Anwendungsbereichs und der Aufnahme des zu lokalisierenden Bildes abhängt. Hier zeigt sich, dass große Zeitspannen eine Herausforderung für alle aktuellen Lokalisierungsmethoden sind, wenn die Bilder in Außenbereichen aufgenommen wurden. Ebenfalls erweist sich die Korrespondenzfindung als schwierig, wenn Karte und Lokalisierungsbild bei unterschiedlichen Wetterbedingungen aufgenommen worden sind. Diese Erkenntnisse weisen auf verbleibende Probleme im Bereich der bodentexturbasierten Lokalisierung hin, die in Zukunft untersucht werden sollten.

Ausführliche Zusammenfassung

Dies ist eine ausführliche Zusammenfassung der zugrundeliegenden Motivation sowie der enthaltenen wissenschaftlichen Beiträge dieser Dissertation.

Motivation

Eine zuverlässige und präzise Eigenlokalisierung ist die Grundlage für viele Aufgaben selbständig handelnder Agenten, wie Roboter oder autonome Fahrzeuge. Die bodentexturbasierte Lokalisierung anhand von visuellen Merkmalen des Bodens ist dabei ein vielversprechender Ansatz mit einigen entscheidenden Vorteilen gegenüber Lokalisierungsmethoden, die beispielsweise visuelle Merkmale aus der Umgebung verwenden. Ein Vorteil besteht darin, dass der Boden in vielen Einsatzgebieten auch über längere Zeiträume (visuell) relativ stabil ist. Dies gilt insbesondere für Einsatzgebiete in Innenräumen und wenn der Agent mit einer eigenen Bodenbeleuchtung ausgestattet ist, sodass die äußerlichen Lichtbedingungen keinen oder nur wenig Einfluss auf die Bildaufnahme haben. Die bodentexturbasierte Lokalisierung ist daher beispielsweise eine Lösung für Anwendungsfälle, in denen andere visuelle Orientierungspunkte regelmäßig ihre Position ändern oder durch Hindernisse verdeckt werden, zum Beispiel auf einem großen Parkplatz oder in einer stark frequentierten Fußgängerzone. Ein weiterer Vorteil besteht darin, dass es mit diesem Ansatz ausreicht den Boden zu betrachten, sodass die Privatsphäre durch den für die Lokalisierung genutzten Sensor nicht beeinträchtigt wird.

Die in dieser Dissertation hauptsächlich betrachtete Aufgabe besteht in der Bestimmung der Position und Orientierung eines Bildes in einer voraufgezeichneten Karte, die wiederum aus einer Vielzahl Referenzbilder besteht, deren Positionen und Orientierungen im Kartenkoordinatensystem zuvor bestimmt worden sind. Der aktuelle Stand der Technik im Bereich der bodentexturbasierten Lokalisierung mit Hilfe einer voraufgezeichneten Karte besteht in der Detektion von charakteristischen visuellen Merkmalen in den kartierten Bodenbildern und dem anschließenden Wiederauffinden der entsprechenden visuellen Merk-

male in den zur Lokalisierung aufgenommenen Bildern. Die Lokalisierung anhand von Bildmerkmalen kann in fünf Schritte unterteilt werden, wobei die ersten drei Schritte jeweils pro Bild durchgeführt werden (inklusive der Referenzbilder), während die beiden weiteren pro Lokalisierungsvorgang durchgeführt werden. (1) Zunächst werden charakteristische Bildbereiche bestimmt, (2) dann wird eine Teilmenge dieser Bildbereiche für die weitere Verarbeitung selektiert, (3) diese Bildbereiche werden dann mit Hilfe eines Verfahrens für die Merkmalsdeskription beschrieben. (4) Nun werden die Deskriptoren der Merkmale des zu lokalisierenden Bildes mit denen der Referenzbilder verglichen, und anhand eines Ähnlichkeitsmaßes wird bestimmt, welche Merkmale miteinander korrespondieren könnten. (5) Die in diesem Prozess gefundenen Merkmalskorrespondenzkandidaten können dann abschließend genutzt werden, um die Position und die Orientierung des zu lokalisierenden Bildes relativ zu den Referenzbildern zu ermitteln.

Auch wenn aktuelle Methoden die Lokalisierungsaufgabe bereits zuverlässig zu lösen scheinen (siehe [Kozak and Alban, 2016, Zhang et al., 2019, Chen et al., 2018]), handelt es sich bei den berichteten Ergebnissen um reine Selbstevaluationen. So fehlte es bisher einerseits an einer vergleichenden Evaluation der vorhandenen Ansätze anhand der gleichen Lokalisierungsprobleme, und andererseits fehlte es an einer ausgiebigen Evaluation möglicher Alternativen und Varianten der vorgeschlagenen Methoden. Insbesondere wurde dem Problem des Rechenaufwandes, beziehungsweise der benötigten Rechenzeit, bisher nur wenig Aufmerksamkeit geschenkt. Gerade dieser Aspekt ist jedoch für den praktischen Einsatz vor allem auf einem kostengünstigen Roboter kritisch, da diesem nur eine geringe Menge an Rechen- und Batteriekapazität zur Verfügung steht, und weil dieser im Echtzeitbetrieb auf eine aktuelle Information der eigenen Position angewiesen ist. Des Weiteren fehlt bisher auch eine systematische Untersuchung verschiedener Ausprägungen des Lokalisierungsproblems, beispielsweise die Lokalisierung mit Hilfe einer bereits vorhandenen groben Schätzung der aktuellen Position. Als weitere bisher nicht untersuchte Ausprägung des Lokalisierungsproblems haben wir die Lokalisierung mit wesentlicher zeitlicher Verzögerung zwischen Kartierung und Zeitpunkt der Lokalisierung identifiziert, wenn diese dazu führt, dass sich der Boden teilweise durch Abtragungen, Verschmutzungen oder Nässe verändert hat.

Diese Dissertation adressiert unter anderem diese offenen Fragestellungen. Die wichtigsten Beiträge dieser Arbeit werden im Folgenden dargelegt.

Beiträge

Diese Dissertation besteht im Wesentlichen aus sechs wissenschaftlichen Beiträgen, von denen vier ([Schmid, Simon, and Mester, 2019, 2020a,b, Schmid, Simon, Radhakrishnan, Frintrop, and Mester, 2022]) in hochrangigen Konferenzproceedings mit einem „peer-review“-System publiziert wurden.

Studie über geeignete Bildmerkmale

Diese Studie [Schmid, Simon, and Mester, 2019] beschäftigt sich damit, einige der am häufigsten verwendeten Merkmalsextraktionsmethoden auf ihre Eignung für die bodentexturbasierte Lokalisierung zu evaluieren.

Das Ziel dieser systematischen Evaluation besteht darin, die für die bodentexturbasierte Lokalisierung am besten geeigneten Verfahren unter den bereits etablierten Verfahren zur Detektion, Selektion, und Deskription von visuellen Merkmalen zu bestimmen. Die Ergebnisse sollen bei der Konstruktion einer bodentexturbasierten Lokalisierungsmethode als Entscheidungshilfe für ein Vorgehen zur Merkmalsextraktion dienen. Gleichzeitig beantwortet diese Arbeit auch die Frage, ob Merkmalsextraktionsverfahren, die für Umgebungsbilder konstruiert wurden, auch für Bodenbilder geeignet sind, oder ob ein grundlegend neues Verfahren für diesen Anwendungsfall entwickelt werden muss.

Zum Zeitpunkt der Veröffentlichung dieser Studie existierten bereits Studien über die Eignung von Merkmalsextraktionsverfahren für die bodentexturbasierte Lokalisierung (siehe [Zhang et al., 2019, Kozak and Alban, 2016, Otsu et al., 2013]). Unsere Studie geht jedoch in mehreren Punkten über diese Arbeiten hinaus. Insbesondere berücksichtigt sie eine größere Anzahl von Merkmalsextraktionsmethoden. Zusätzlich zur Merkmalsdetektion und Merkmalsdeskription berücksichtigt sie noch die Merkmalsselektion, und es werden zusätzliche Erfolgsmetriken evaluiert, sodass die Vor- und Nachteile der Methoden detaillierter untersucht werden können. Des Weiteren ist unsere Studie die erste, die sowohl die Eignung für die inkrementelle Bild-zu-Bild Lokalisierung, als auch für die absolute kartenbasierte Lokalisierung untersucht.

Zunächst untersuchen wir in dieser Studie die Merkmalsdetektion in Kombination mit der Merkmalsselektion. Dabei geht es darum, eine Menge von Bildbereichen zu bestimmen, die zuverlässig in unabhängig voneinander verarbeiteten Bildern mit überlappender Bodenabdeckung wiedergefunden werden. Das heißt, dass in dem überlappenden Bereich zweier Bodenbilder möglichst die gleichen Bildbereiche bestimmt werden sollten. Um dies zu untersuchen,

verwenden wir synthetische Bildtransformationen. Dazu wird ein Bild beispielsweise gedreht, mit Rauschen versehen oder es wird durch eine Gammakorrektur eine Beleuchtungstransformation durchgeführt. Ein Vorteil dieser synthetischen Transformationen besteht darin, dass die untersuchten Methoden systematisch auf ihre Robustheit gegenüber den jeweiligen Transformationen untersucht werden können. Zusätzlich zu den synthetisch transformierten Bildpaaren untersuchen wir Paare von separat aufgenommenen überlappenden Bodenbildern. Hierbei betrachten wir Bildpaare, die in direkter Sequenz nacheinander aufgenommen wurden, um die Methoden auf ihre Eignung für die Aufgabenstellung der inkrementellen Lokalisierung zu untersuchen, und wir betrachten unabhängig voneinander aufgenommene Bilder (unabhängige Aufnahmezeitpunkte und unabhängige Bildorientierungen), um die Methoden für die kartenbasierte Lokalisierung zu untersuchen.

Unsere Ergebnisse zeigen, dass die vorhandenen Methoden zur Merkmalsextraktion grundsätzlich für die bodentexturbasierte Lokalisierung geeignet sind. So konnten wir für die verschiedenen Lokalisierungsprobleme jeweils Kombinationen aus Merkmalsdetektor, Merkmalsselektor, und Merkmalsdeskriptor finden, die nahezu optimale Ergebnisse erzielen. Sie ermöglichen uns also eine ausreichend große Anzahl korrekter Bildmerkmalskorrespondenzen zu finden, während nur eine geringe Anzahl inkorrekter Korrespondenzen erzeugt wird. Dabei bestätigt unsere Studie im Wesentlichen die Schlussfolgerungen anderer Untersuchungen dazu, welche Methoden für Bodenbilder geeignet sind. Dementsprechend ergibt sich aus unserer Studie, dass es für die untersuchten Aufgabenstellungen keinen dringenden Bedarf für ein grundlegend neues Verfahren zur Merkmalsextraktion aus Bodenbildern gibt.

Lokalisierungsmethode basierend auf Identitätsabgleich und kompakten binären Deskriptoren

In diesem Beitrag [Schmid, Simon, and Mester, 2020a] präsentieren wir eine neue Methode für die bodentexturbasierten Lokalisierung, welche insbesondere mit dem Ziel entwickelt wurde, dass sie schneller berechnet werden kann als andere aktuelle Methoden um für den Echtzeiteinsatz geeignet zu sein.

Wir adaptieren hierzu Micro-GPS, eine Methode von Zhang et al. [2019], welche durch den Einsatz einer geometrischen Plausibilitätsprüfung zum Aussortieren von inkorrekt vorgeschlagenen Merkmalskorrespondenzen gekennzeichnet ist. Außerdem setzt die Methode auf eine effiziente Suchstruktur, um für jedes Merkmal des Lokalisierungsbildes das Merkmal aus den Referenzbildern mit

ähnlichstem Deskriptor ausfindig zu machen. Da die Erstellung der Suchstruktur mit erheblichem Rechenaufwand verbunden ist, wird diese offline, also vor der Lokalisierung, global erstellt, sodass gleichzeitig nach Korrespondenzen im gesamten Anwendungsgebiet gesucht wird. Diese Suchstruktur ersetzen wir durch ein neues Verfahren zur Korrespondenzfindung: den Identitätsabgleich. Hierbei werden alle Merkmale mit identischem Deskriptor als mögliche Korrespondenzen in Betracht gezogen. Wir schlagen eine kosteneffiziente Implementierung des Identitätsabgleichs mit Hilfe von kompakten binären Deskriptoren vor. Dabei beobachten wir bei der Anwendung dieser Technik, dass jedem Merkmal aus dem Lokalisierungsbild in der Regel eine große Anzahl möglicher Korrespondenzen aus den Referenzbildern zugewiesen wird. Wobei nur sehr wenige dieser vorgeschlagenen Korrespondenzen als korrekte Korrespondenzen angesehen werden können. Um diese wiederum von den inkorrekten Korrespondenzen zu trennen, erweist sich die geometrische Plausibilitätsprüfung von Zhang et al. als äußerst nützlich.

Wir führen eine ausführliche Evaluation der vorgeschlagenen Lokalisierungsmethode durch. Dabei vergleichen wir diese auch mit der Originalimplementation von Micro-GPS, sowie zwei weiteren von uns nachimplementierten Methoden: Ranger [Kozak and Alban, 2016] und StreetMap [Chen et al., 2018]. In einem ersten Experiment testen wir, ob die Methoden in der Lage sind, sich auf Gebieten mit Größen zwischen $17,70\,\mathrm{m}^2$ bis $41,76\,\mathrm{m}^2$, die von 2014 bis 4043 Referenzbildern abgedeckt werden [Zhang et al., 2019], erfolgreich lokalisieren können. Wir finden heraus, dass unsere Methode häufiger eine korrekte Lokalisierung zustande bringt, als dies mit der Vergleichsmethode der Fall ist. Anschließend führen wir eine Evaluation durch, bei der eine ungefähre Position des zu lokalisierenden Bildes bereits bekannt ist. Dadurch ist es bei der Lokalisierung möglich, den berücksichtigten Suchradius in der Karte einzuschränken, indem lediglich die Referenzbilder in der örtlichen Umgebung der geschätzten Position in Betracht gezogen werden. Aus der Untersuchung dieser Aufgabe ergibt sich, dass unsere Methode etwas seltener in der Lage ist, sich korrekt zu lokalisieren als Ranger und StreetMap. Es zeigt sich allerdings, dass unsere Methode durch die Verwendung des Identitätsabgleichs für die Korrespondenzfindung einen Laufzeitvorteil gegenüber den anderen Methoden hat. Dies gilt insbesondere im Vergleich mit Micro-GPS, welches durch die Verwendung der globalen Suchstruktur für die Korrespondenzfindung nicht in der Lage ist, die Menge der zur Lokalisierung berücksichtigten Referenzbilder einzuschränken.

Mit unserer Methode lässt sich die Position und Orientierung des Lokalisierungsbildes schnell bestimmen, jedoch gilt dies nur, wenn die Merkmalsex-

traktion beispielsweise mit Hilfe einer schnellen Grafikkarte durchgeführt wird, da ein rechenaufwändiges Verfahren zur Merkmalsdetektion eingesetzt wird (SIFT [Lowe, 2004]). In einer Folgeaktivität [Schmid, Simon, and Mester, 2020b] präsentieren wir eine Variante unserer Lokalisierungsmethode, die in der Lage ist den Rechenaufwand noch einmal weiter zu reduzieren, sodass ein rein CPU-basierter Echtzeiteinsatz ermöglicht wird.

Dazu ersetzen wir die Verwendung eines Merkmalsdetektors mit einem Stichprobenverfahren, bei dem unabhängig vom eigentlichen Bildinhalt beliebige Bildausschnitte für die Merkmalsbildung verwendet werden. Die Bildausschnitte können dabei entweder zufällig oder nach einem festgelegten Muster bestimmt werden. Ein Vorteil, der sich aus dieser Technik ergibt, besteht darin, dass die Rechenkosten für die Merkmalsdetektion praktisch vollständig eliminiert werden. Als Nachteil beobachten wir hingegen, dass eine größere Anzahl Merkmale pro Bild verwendet werden muss, um eine ähnliche Lokalisierungsleistung zu erzielen wie mit den klassischen Merkmalsdetektoren. Daher erhöht die Verwendung des Stichprobenverfahrens den Speicherbedarf.

Unsere Ergebnisse zeigen, dass sowohl unsere Lokalisierungsmethode, als auch Ranger [Kozak and Alban, 2016] und StreetMap [Chen et al., 2018] ähnlich gute Leistung mit unserem Stichprobenverfahren, wie mit den klassischen Merkmalsdetektoren, erzielen können. Hinsichtlich des Rechenaufwands profitiert dabei unsere Methode am stärksten, da sie durch die Verwendung des Identitätsabgleichs sehr günstig mit der Anzahl der berücksichtigten Merkmale skaliert. Insgesamt erreichen wir mit dieser Variante unserer Methode das Ziel einer rein CPU-basierten echtzeitfähigen Lokalisierungsmethode.

Modellbasierte Vorhersage

Eine Erkenntnis aus unseren vorherigen Beiträgen besteht darin, dass die Lokalisierungsleistung der untersuchten Methoden stark von der verwendeten Parametrisierung, insbesondere die der Merkmalsextraktionsmethode, abhängt. Aufgrund der Größe der Parameterräume der evaluierten Lokalisierungsmethoden ist das finden einer geeigneten Parametrisierung jedoch rechen- beziehungsweise zeitaufwändig. Daher stellen wir im Beitrag [Schmid, Simon, and Mester, 2021] ein Vorhersagemodell vor, das wir in einem automatisierten Parameteroptimierungsverfahren einsetzen können, um mit wenig Rechenaufwand geeignete Parameter zu ermitteln.

Unser Vorhersagemodell kann dafür genutzt werden, eine mögliche Parametrisierung zu evaluieren, ohne dass diese dafür vollständig getestet werden

muss. Dazu wird die Lokalisierungsmethode entsprechend parametrisiert und lediglich auf einigen wenigen Testbildern ausgewertet. Anschließend werden wichtige sich dabei ergebende Kennzahlen, wie die Anzahl korrekter und inkorrekter gefundener Korrespondenzen, genutzt, um zu prädizieren, wie erfolgreich die Lokalisierungsmethode wäre, wenn nicht nur wenige Testbilder, sondern alle Referenzbilder der Karte berücksichtigt worden wären.

Unsere anschließende Evaluation zeigt, dass das Modell die Lokalisierungserfolgsrate ausreichend präzise vorhersagen kann, sodass wir es in einem einfachen automatisierten Parametrisierungsverfahren für die Auswertung von in Betracht gezogenen Parametrisierungen nutzen können. Wir finden mit diesem Verfahren innerhalb einiger Stunden Parametereinstellungen für unsere Lokalisierungsmethode, mit denen wir uns ähnlich oft erfolgreich lokalisieren, wie mit unseren aufwändig manuell bestimmten Parametern. Darüber hinaus konnte die Rechenzeit unserer Lokalisierungsmethode mit den automatisiert gefundenen Parametern noch einmal deutlich reduziert werden. Während des Einsatzes des Parametrisierungsverfahrens werden hunderte Parametrisierungen ausgewertet. Die Auswertung für eine einzelne Parametrisierung dauert mit unserem Modell etwa 20 Sekunden, während dies ohne die Verwendung des Modells über 55 Minuten benötigt.

Bildabgleich für die Suche überlappender Bodenbilder

Ein erfolgversprechender Ansatz zur Verbesserung der Lokalisierungserfolgsrate ist die Reduzierung der Anzahl der bei der Lokalisierung berücksichtigten Referenzbilder, die keine Überlappung mit dem Lokalisierungsbild haben. Eine einfache Möglichkeit dies zu erreichen, besteht darin, nur die Referenzbilder aus der tatsächlichen örtlichen Umgebung des Lokalisierungsbildes zu berücksichtigen. Dies ist beispielsweise möglich, wenn die ungefähre Position bereits vor der Lokalisierung bekannt ist. Ein alternativer Ansatz, der auch ohne eine ungefähr bekannte Position funktioniert, besteht darin, einen Bildabgleich durchzuführen, mit dem die mit dem Lokalisierungsbild überlappenden Referenzbilder identifiziert werden können.

Ein solcher Bildabgleich wird bereits von der Lokalisierungsmethode StreetMap [Chen et al., 2018] eingesetzt, wenn beispielsweise bei der initialen Lokalisierung keine ungefähre Position bekannt ist. Hier wird ein sogenannter Bag-of-Words (BoW) Ansatz [Galvez-López and Tardos, 2012] verwendet. Dabei handelt es sich um einen aggregierten Bilddeskriptor, der mit Hilfe der Deskriptoren der aus dem Bild extrahierten Merkmale berechnet wird.

Unser Beitrag [Radhakrishnan, Schmid, Scholz, and Schmidt-Thieme, 2021] besteht in einem neuen, auf tiefen künstlichen neuronalen Netzen basierenden, Verfahren für diesen Bildabgleich. Dabei lernt das Netz vorherzusagen, wie groß die geometrische Überlappung zweier Bodenbilder ist, indem es für diese Bilder Deskriptoren generiert, deren euklidischer Abstand umgekehrt proportional zur Überlappung wächst. Dementsprechend werden die Gewichte des Netzes so angepasst, dass die Abstände der Bilddeskriptoren der am stärksten überlappenden Bilder am kleinsten wird. Anschließend können diese Bilddeskriptoren für die Lokalisierung verwendet werden, indem die Referenzbilder mit den Deskriptoren mit geringstem Abstand zum Deskriptor des Lokalisierungsbildes verwendet werden.

In unserer experimentellen Evaluation fixieren wir die Anzahl berücksichtigter Referenzbilder mit kleinsten Deskriptorabständen zum Deskriptor des Lokalisierungsbildes. Dabei zeigt sich, dass mit unserer Methode ein wesentlich größerer Anteil dieser berücksichtigten Referenzbilder tatsächlich mit dem Lokalisierungsbild überlappt als dies mit der BoW-Methode der Fall ist. Dies gilt insbesondere für die schwieriger zu identifizierenden Referenzbilder mit geringer Überlappung (kleiner 40%) mit dem Lokalisierungsbild. Es zeigt sich auch, dass die Erfolgsrate unserer Lokalisierungsmethode deutlich gesteigert werden kann, wenn nicht alle Referenzbilder bei der Lokalisierung in Betracht gezogen werden, sondern nur jene, die von unserer Bildabgleichsmethode extrahiert wurden. Die dabei erreichte Erfolgsrate ist zudem höher, als die Erfolgsrate bei Verwendung der BoW-Methode.

Die *HD Ground* Datenbasis

Die systematische Evaluation ist ein essentieller Bestandteil der Entwicklung neuer Lokalisierungsmethoden. Während eine Evaluation im Realbetrieb dabei Aufschluss über die tatsächliche Anwendbarkeit der Methode liefern kann, ist diese Art der Evaluation mit erheblichen Aufwand verbunden. Zudem ist zu erwarten, dass sich die Ergebnisse von Durchlauf zu Durchlauf unterscheiden werden. Für eine einfache und reproduzierbare Evaluation kann stattdessen auf eine voraufgezeichnete Datenbasis gesetzt werden. Damit die Evaluation dennoch aussagekräftige Ergebnisse liefern kann, sollte diese Datenbasis möglichst umfangreich sein und die relevanten Anwendungsfälle abdecken.

Für die bisher beschriebenen Beiträge haben wir die Micro-GPS Datenbasis [Zhang et al., 2019] verwendet. Nach unserem Wissen handelt es sich hierbei bisher um die einzige öffentlich verfügbare Datenbasis, die für die Evaluation

bodentexturbasierter Lokalisierungsmethoden geeignet ist. In unseren vorherigen Beiträgen zeigte sich jedoch, dass für viele Lokalisierungsaufgaben alle evaluierten Methoden nahezu optimale Lokalisierungserfolgsraten erzielen. Daher stellt sich die Frage, ob diese Datenbasis tatsächliche alle relevanten Herausforderungen der bodentexturbasierten Lokalisierung abdeckt. So ist es mit der Micro-GPS Datenbasis beispielsweise nicht möglich, auf systematische Weise zu evaluieren, wie sich die Lokalisierungsleistung verhält, wenn zwischen dem Zeitpunkt der Kartierung des Anwendungsbereichs und dem Zeitpunkt der Bildaufnahme des Lokalisierungsbildes größere Zeitintervalle liegen. In der Praxis ist dies relevant, da die Kartierung mit erheblichem Aufwand verbunden sein kann, sodass man diese nicht regelmäßig wiederholen möchte. Weitere Fragestellungen bestehen darin, wie sich die Lokalisierungserfolgsraten auf größeren Anwendungsflächen und anderen Bodentexturen verhalten, als den in der Micro-GPS Datenbasis enthaltenen Anwendungsflächen.

Unser Beitrag besteht in einer neuen Datenbasis für die bodentexturbasierte Lokalisierung [Schmid, Simon, Radhakrishnan, Frintrop, and Mester, 2022]. Wir nennen sie die *HD Ground* Datenbasis. Des Weiteren stellen wir unsere Aufnahmeplattform vor und erläutern die wesentlichen Aspekte, die bei ihrer Konstruktion berücksichtigt wurden. Im Vergleich mit der Micro-GPS Datenbasis enthält unsere Datenbasis eine größere Vielfalt verschiedener Bodentypen und deutlich mehr Bilder, die den Boden mit höherer Auflösung und mit wesentlich weniger Bewegungsunschärfe abbilden. Die größte Anwendungsfläche unserer Datenbasis ist mit $106,12\,\mathrm{m}^2$ deutlich größer als die der Micro-GPS Datenbasis mit $40,76\,\mathrm{m}^2$. Im Vergleich deckt unsere Datenbasis eine mehr als doppelt so große Gesamtfläche ab. Darüber hinaus ermöglicht die HD Ground Datenbasis die zuvor beschriebene systematische Untersuchung der Lokalisierungsleistung bei größer werdenden Zeitintervallen zwischen Kartierung und Lokalisierung. Dafür haben wir über einen Zeitraum von 24 Wochen für vier Anwendungsflächen die gleichen Teststrecken im wöchentlichen Rhythmus aufgenommen. Außerdem ermöglicht die HD Ground Datenbasis die Untersuchung eines „Teach-and-Repeat“-Anwendungsfalls, bei dem die Aufgabe darin besteht, einen einmalig eingelernten Pfad autonom in beide Richtungen abfahren zu können.

Teil dieses Beitrages ist auch die Fortführung der Evaluationen einiger zuvor beschriebener Beiträge. Dabei finden wir heraus, dass die Erfolgsraten der initialen Lokalisierung auf den von uns kartierten Anwendungsflächen teilweise deutlich geringer sind, als dies mit der Micro-GPS Datenbasis der Fall ist. Insbesondere auf den Anwendungsflächen im Freien hängt die Erfolgsrate hierbei

von dem Zeitintervall zwischen Kartierung und Lokalisierung ab, wobei die Erfolgsrate mit größer werdenden Abständen teilweise deutlich abnimmt. Für die auf unserer Datenbasis entsprechend wesentlich größere Herausforderung der initialen Lokalisierung erweist sich die Verwendung unseres auf tiefen künstlichen neuronalen Netzen basierenden Bildabgleichverfahren für die Suche überlappender Bodenbilder als vorteilhaft. Im Vergleich mit dem BoW-Verfahren, und vor allem im Vergleich mit einer Anwendung ohne Reduzierung der berücksichtigten Referenzbilder, ergeben sich deutlich bessere Erfolgsraten unserer Lokalisierungsmethode. Ebenfalls beobachten wir höhere Erfolgsraten im „Teach-and-Repeat“-Anwendungsfall.

Der Vergleich der Lokalisierungsmethoden Ranger [Kozak and Alban, 2016] und StreetMap [Chen et al., 2018] mit unseren Lokalisierungsmethoden zeigt auf der HD Ground Datenbasis, dass Ranger und StreetMap tendenziell höhere Erfolgsraten erzielen. Zudem zeigen wir, dass diese Methoden ebenfalls sehr kurze Rechenzeiten realisieren können, wenn die Bildauflösung vor der Verarbeitung deutlich reduziert wird. Es ergibt sich jedoch weiterhin eine vorteilhafte Skalierung der Rechenzeit unserer Methoden für größere Anzahlen berücksichtigter Referenzbilder, wie sie sich beispielsweise bei ungenauer Kenntnis der aktuellen Position ergeben können.

Danksagung

Ich bedanke mich bei

- meiner Frau, die immer an meiner Seite stand, auch wenn wir nicht beisammen sein konnten.
- meinen Eltern Christine und Bem, sowie meiner Schwester Julia, die mich geprägt haben und auf die ich mich immer verlassen kann.
- meinem Betreuer Dr. Stephan Simon, der mich in jeder Hinsicht unterstüzt und gefördert hat, und mit dem ich viele konstruktive und spannende Diskussionen führen durfte.
- meinem Doktorvater Prof. Dr. Rudolf Mester, der trotz eines vollen Terminkalenders stets für eine Rücksprache zur Verfügung stand.
- Prof. Dr. Simone Frintrop für ihre Tipps zum Schreiben und Publizieren wissenschaftlicher Arbeiten und ihre wertvolle Unterstützung.
- meinen Kollegen, insbesondere Moritz, Annika, Charlotte, Marie, Christian, Maria, und Holger, die mich mit Korrekturlesen, Tipps, Diskussionen, und Expertise unterstützt haben.
- meinen Studenten Raaghav, Michael, Haljan, Sharang, Jan, Rene, Malte, und Thomas für ihren Einsatz und ihre Begeisterung für die von uns gemeinsam bearbeiteten Projekte, sowie für ihre vielen tollen Fragen.
- meinen Vorgesetzten an der Robert Bosch GmbH für ihren Glauben an meine Fähigkeiten und ihre Unterstützung.

Contents

Acronyms

GNSS	Global Navigation Satellite System
GTBL	Ground Texture Based Localization
ANN	Approximate Nearest Neighbor
BoW	Bag of Words
RANSAC	RAndom SAmple Consensus
DoG	Difference-of-Gaussian
NMS	Non-Maximum Suppression
ANMS	Adaptive Non-Maximum Suppression
IoU	Intersection over Union
PCA	Principal Component Analysis
SD	Standard Deviation
CSR	Complete Spatial Randomness
CNN	Convolutional Neural Network
DML	Deep Metric Learning
ICP	Iterative Closest Point
GPU	Graphics Processing Unit

Notation

General

$\mathcal{A}$	A set.
f	A function.
a	A scalar variable.
$\boldsymbol{v}$	A vector.
$\boldsymbol{t}$	A translation vector.
$\mathbf{M}$	A matrix.
$\mathbf{R}$	A rotation matrix.
$[\mathbf{R}\|\boldsymbol{t}]$	A 2D Euclidean transformation, consisting of a rotation $\mathbf{R}$ and a translation $\boldsymbol{t}$.

Localization

$\mathcal{R}$	The set of reference images, covering the application area.
$\mathrm{r} \subset \mathcal{R}$	A reference image.
$\mathcal{T}_{\mathcal{R}}$	Poses of the reference images $\mathcal{R}$.
$\mathrm{f_r}$	Image processing function extracting relevant information from the reference images $\mathcal{R}$.
m	Mapping function that constructs a map from the given reference images $\mathcal{R}$ and their corresponding poses $\mathcal{T}_{\mathcal{R}}$.
M	The map created by m that stores the extracted information of the reference images $\mathcal{R}$.
q	The query image which is to be localized.
$\mathrm{f_q}$	Image processing function extracting relevant information from the query image q.
g	The query image pose estimation function.
$[\mathbf{R}\|\boldsymbol{t}]_{\mathrm{q}}^{\mathrm{M}}$	The actual pose of the query image q in the coordinate system of the map M.
$[\mathbf{R}_{\mathrm{est}}\|\boldsymbol{t}_{\mathrm{est}}]_{\mathrm{q}}^{\mathrm{M}}$	An estimate of the query image pose $[\mathbf{R}\|\boldsymbol{t}]_{\mathrm{q}}^{\mathrm{M}}$ from the localization function g.
$[\mathbf{R}_{\mathrm{p}}\|\boldsymbol{t}_{\mathrm{p}}]_{\mathrm{q}}^{\mathrm{M}}$	The prior, an estimation of the current pose $[\mathbf{R}\|\boldsymbol{t}]_{\mathrm{q}}^{\mathrm{M}}$ that is already available during localization.
d_{p}	The expected accuracy of the prior in form of the maximum distance of the prior pose estimate to the actual query image pose.
d_{t}	Euclidean distance threshold to determine whether the query image pose estimate $[\mathbf{R}_{\mathrm{est}}\|\boldsymbol{t}_{\mathrm{est}}]_{\mathrm{q}}^{\mathrm{M}}$ is considered to be correct.
o_{t}	Orientation threshold of the absolute angle difference to determine whether the query image pose estimate $[\mathbf{R}_{\mathrm{est}}\|\boldsymbol{t}_{\mathrm{est}}]_{\mathrm{q}}^{\mathrm{M}}$ is considered to be correct.

Feature-Based Localization

$\mathcal{M}$	The set of feature matches, i. e. proposed feature correspondences potentially containing incorrect matches (outliers).
$\mathcal{I} \subset \mathcal{M}$	The set of inlier matches, that are considered to represent correct correspondences.
$\mathcal{O} \subset \mathcal{M}$	The set of outlier matches, $\mathcal{M} = \mathcal{I} \cup \mathcal{O}$.
$\mathcal{F}_\mathcal{R}$	The set of reference features, i. e. the union of all features extracted from the reference images.
n_r	The number of extracted features per reference image.
$\mathcal{F}_\mathrm{q}$	The set of query image features.
$m \in \mathcal{M}$	A proposed feature match $m = (f_q \in \mathcal{F}_\mathrm{q}, f_r \in \mathcal{F}_\mathcal{R})$.

Probability Theory

$\mathcal{A}$	An event.
$\Pr[\mathcal{A}]$	Probability of the event $\mathcal{A}$.
$\Pr[\mathcal{A} \cap \mathcal{B}]$	The joint probability of the events $\mathcal{A}$ and $\mathcal{B}$.
$\Pr[\mathcal{A}\|\mathcal{B}]$	Probability of the event $\mathcal{A}$, given $\mathcal{B}$.
X	A random variable.
x	Value that the random variable X takes.
$\Pr[X = x]$	Probability of X taking the value x.
$\mathrm{B}(i\|p, n)$	The probability of observing i successes in n independent Bernoulli trials, each with a success probability of p.

Modeling the Localization Success Rate

$\mathcal{V}$	The set of voting grid cells that received at least one vote during the voting procedure.
$\mathcal{V}_\mathcal{I} \subset \mathcal{V}$	The set of voting grid cells that received at least one inlier vote during the voting procedure.
$N^v_\mathcal{M}$	The random variable that represents the number of votes cast onto $v \in \mathcal{V}$.
$N^v_\mathcal{I}$	The random variable denoting the number of inliers in $N^v_\mathcal{M}$ for $v \in \mathcal{V}$.
$N^v_\mathcal{O}$	The random variable denoting the number of outliers in $N^v_\mathcal{M}$ for $v \in \mathcal{V}$.
v_p	The voting peak, i. e. the voting cell that received most votes, $v_p = \{v \in \mathcal{V} \| N^v_\mathcal{M} = \max_{v' \in \mathcal{V}} N^{v'}_\mathcal{M}\}$.
$p_\mathrm{out\_vote}$	Probability of an outlier match $m \in \mathcal{O}$ casting a vote on a particular voting cell $v \in \mathcal{V}$, (which is the same for any voting cell $v \in \mathcal{V}$).
$p^v_\mathrm{in\_vote}$	Probability of query feature $f_q \in \mathcal{F}_\mathrm{q}$ of generating one inlier vote on $v \in \mathcal{V}$.

1 Introduction

Contents of this chapter were partially published in [Schmid, Simon, and Mester, 2019], [Schmid, Simon, and Mester, 2020a], and [Schmid, Simon, Radhakrishnan, Frintrop, and Mester, 2022].

Accurate self-localization capabilities are required for nearly all robotics tasks [Thrun et al., 2005, chap. 7]. In particular, it is a prerequisite for autonomous agents to perform tasks such as freight and passenger transport [Cornick et al., 2016] and it is important for the use of robotic vacuum cleaners and social robots [Chen et al., 2014]. Available solutions such as Global Navigation Satellite System (GNSS) for outdoor applications are not able to reliably provide accurate positioning in urban environments [Cornick et al., 2016], and systems for indoor applications such as Ultra Wideband require installation of costly infrastructure [Chen et al., 2018, Fang et al., 2009]. Visual localization using environmental landmarks can achieve centimeter precise localization in some indoor applications, but might suffer from occlusions of the perceived surrounding and can deviate meters from the correct position in outdoor scenarios [Mur-Artal and Tardós, 2017].

Ground texture based localization approaches using a single downward-facing camera, on the other hand, present promising results for a cost-effective solution for reliable localization with centimeter accuracy [Zhang et al., 2019, Chen et al., 2018]. Suitable texture types like concrete, asphalt, or carpet are prevalent and remain sufficiently stable in most application areas of autonomous agents [Zhang et al., 2019, Kelly et al., 2007]. An agent that uses the ground instead of surrounding landmarks to localize itself has several advantages:

- it works in dynamic environments with frequently changing surrounding;
- it works with an occluded surrounding, e. g. in a busy pedestrian zone;
- it observes only the ground reducing privacy concerns;
- if the agent actively illuminates the recording area, localization becomes robust to changes in exterior lighting conditions.

We examine the state of the art of ground texture based localization methods,

and their key success factors. In addition, we build on the techniques we find to be most successful to address real-world application issues, such as the real-time applicability, efficient parametrization, and the availability of positioning information in any situation: without having any knowledge about the agent's whereabouts, as well as with approximate positioning information being available.

For the development of our own localization method, we propose a novel feature matching technique that we call *identity matching* that matches only those pairs of local visual features which have bit-identical descriptors. In order for this case to occur with a sufficiently high probability, we employ compact binary descriptors that describe image patches with only a small number of bits, e. g. 15 in our first implementation. A major advantage of this approach is its computational efficiency. On the one hand, this approach allows us to use fast-to-compute binary descriptors, and, on the other hand, comparing for identity presents a particularly efficient way of matching.

1.1 Problem Statement

This dissertation is mainly concerned with map-based *absolute localization*, i. e. given a pre-constructed map we determine the robot positioning in the map-coordinate system. Consider an agent such as an autonomous robot with restricted operational area, e. g. a warehouse robot, equipped with a downward-facing camera. To be able to take on tasks and navigate in its operational area, the robot needs a map and needs to be able to localize itself within that map, i. e. the robot needs to determine its own pose in the map-coordinate system. Here, the term *pose* refers to the combination of the position (x- and y-coordinates), as well as the orientation of the robot.

Figure 1.1 visualizes absolute localization. It is a two step process, in which the pose of a query image is estimated in respect to the reference images of a previously created map. In distinction to that *incremental relative localization*, as visualized in Figure 1.2, is the task of determining the pose of one image directly in respect to another one, which can be done for any two overlapping ground images.

We assume to have a vertically oriented pinhole camera with constant distance to the ground, which is considered to be locally flat. Accordingly, camera poses are in the form of standard Euclidean transformations $[\mathbf{R}|\boldsymbol{t}]$ of rotation $\mathbf{R}$ and translation $\boldsymbol{t}$ in two dimensions.

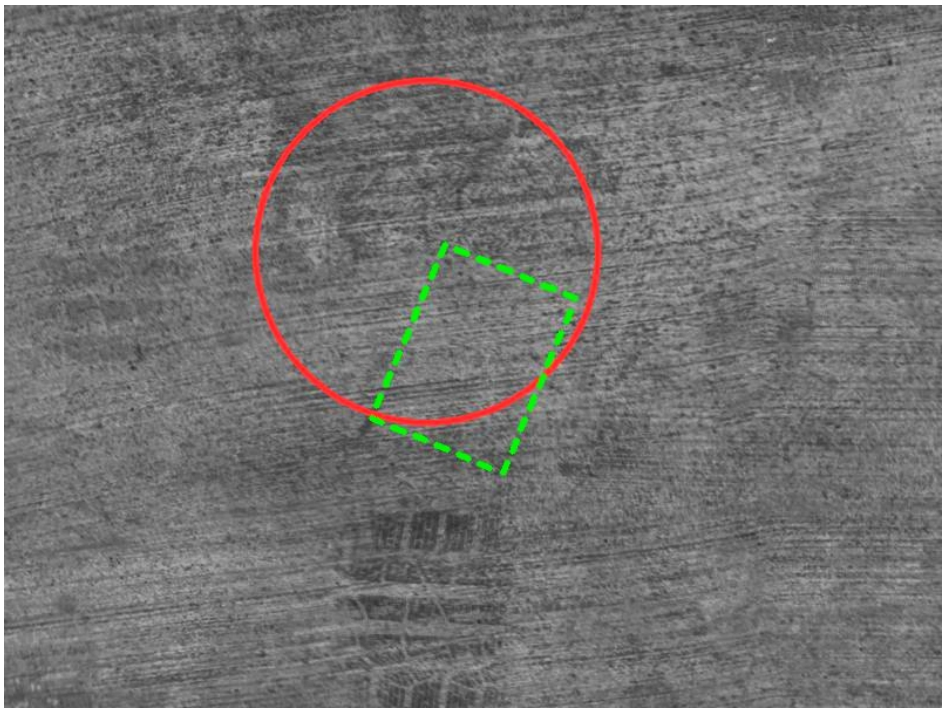
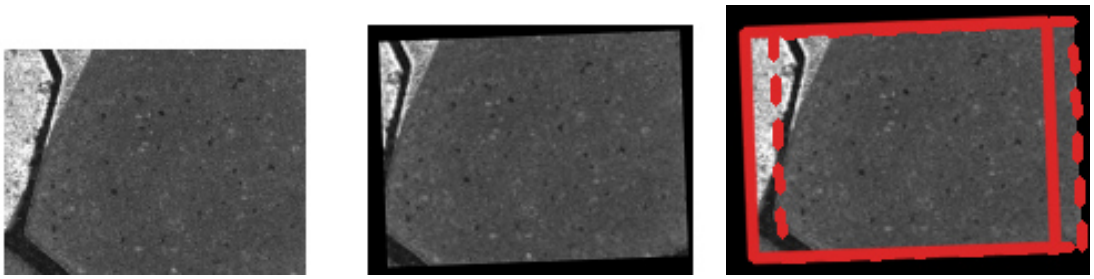


Figure 1.1: Visualization of the map-based absolute localization task, where the query image pose is estimated in respect to the reference images. Images are taken from the garage concrete dataset of our HD Ground Database [Schmid et al., 2022]. The background presents the mapped reference images as an image stitching, and the true query image pose is shown by a green dashed border around the image. Depending on the localization mode, we are either considering all reference images for potential overlap with the query image (*global map-based localization*), or an available query image pose estimate allows to consider only the reference images of a local area (*local map-based localization*). Here, we assume to have a pose estimate available according to which the query image is located in the red circle. It is useful to consider only the closest reference images to the estimated query image position for potential overlap, e. g. one could use all images with overlap to the circle area.

We distinguish between three problems to be solved for absolute localization:

1. The initial **scanning** of the area of operation; the agent explores the environment, gathering observations in form of ground images, and estimates their corresponding poses in the world. The estimated pose of any reference image in the map-coordinate system is expected to be locally consistent with the estimated poses of its neighboring images. As a result of this phase, we obtain a set of *reference images* $\mathcal{R}$ with known poses $\mathcal{T}_{\mathcal{R}}$ in the map-coordinate system.

2. The **creation of a map** data structure from the recorded data; in order to use the reference images for localization, they are processed to create a



(a) The query image (b) The reference image (c) The image stitching

Figure 1.2: Visualization of the incremental relative localization task. Images are taken from the parking place cobblestone dataset of our HD Ground Database [Schmid et al., 2022]. Here, the pose of the query image (a) is estimated in respect to a single reference image (b). Figure (c) shows the estimated relative pose of the query image to the reference image as a joint image stitching of the two images, where the reference image is highlighted with a dashed border line and the query image with a continuous border line.

map data structure M. This could mean for example that visual features are extracted from the images and systematically stored in a data structure that allows for efficient matching with the features found in the image which is to be localized. In this dissertation the initial scanning and the creation of the map are performed sequentially after each other, but in practice they could be performed simultaneously. In our case, the initial scanning, including the estimation of reference image poses is performed only once, as it is independent of the further processing steps. Accordingly, the same set of reference image poses $\mathcal{T}_\mathcal{R}$ is provided to all evaluated localization methods. The procedure for map creation, on the other hand, depends on the employed localization method.

3. The subsequent self-**localization** within the mapped area; once a map is available, it can be used to localize independently recorded *query images*, e. g. by searching the map for visual features that correspond to the features of the query image q. The goal is to estimate the actual pose $[\mathbf{R}|\boldsymbol{t}]_\mathrm{q}^\mathrm{M}$ of the query image in the map coordinate system as accurately as possible. We differentiate between *global localization* without an available estimate of the current pose and *local localization* with available prior pose estimate. These cases are differentiated dependent on the expected position error d_p of the prior pose estimate $[\mathbf{R}_\mathrm{p}|\boldsymbol{t}_\mathrm{p}]_\mathrm{q}^\mathrm{M}$. In the following, we will use the term *prior* to refer to an available estimate of the query image pose. A localization algorithm might treat the cases of global localization with $d_\mathrm{p} = \infty$ and local localization with $d_\mathrm{p} \in \mathbb{R}$ separately or it may have a common approach to both cases. The output of the localization step

is an estimate of the query image pose $[\mathbf{R}_{\text{est}}|\boldsymbol{t}_{\text{est}}]_{\text{q}}^{\text{M}}$, which is considered correct if it is closer to the actual pose $[\mathbf{R}|\boldsymbol{t}]_{\text{q}}^{\text{M}}$ than a threshold distance d_{t} and if the absolute angle between the two Euclidean transformations is smaller than an orientation threshold o_{t}.

We formalize the three introduced problems:

Problem 1 (Scanning) *Create a set of observations of the environment in form of ground images $\mathcal{R}$ (the reference images), and estimate their corresponding poses $\mathcal{T}_{\mathcal{R}}$ in a common coordinate system.*

Problem 2 (Map Creation) *Given a set of reference images $\mathcal{R}$ and their poses $\mathcal{T}_{\mathcal{R}}$, process the images to extract relevant information using an image processing function f_{r}. Subsequently, construct a map M that stores the extracted information efficiently using a mapping function m and the pose estimates $\mathcal{T}_{\mathcal{R}}$:*

$$\text{M} = \text{m}(\text{f}_{\text{r}}(\mathcal{R}), \mathcal{T}_{\mathcal{R}}). \tag{1.1}$$

Problem 3 (Localization) *Given a map M, an observation of the environment in form of a query image q, an image processing function f_{q}, and a localization prior $[\boldsymbol{R}_{\text{p}}|\boldsymbol{t}_{\text{p}}]_{\text{q}}^{\text{M}}$ with its expected accuracy $d_{\text{p}} \in \mathbb{R} \cup \{\infty\}$, estimate the pose $[\boldsymbol{R}|\boldsymbol{t}]_{\text{q}}^{\text{M}}$ of the query image using a pose estimation function g:*

$$[\boldsymbol{R}_{\text{est}}|\boldsymbol{t}_{\text{est}}]_{\text{q}}^{\text{M}} = \text{g}(\text{f}_{\text{q}}(\text{q}), \text{M}, [\boldsymbol{R}_{\text{p}}|\boldsymbol{t}_{\text{p}}]_{\text{q}}^{\text{M}}, d_{\text{p}}). \tag{1.2}$$

1.2 State of the Art

The current state of the art [Kozak and Alban, 2016, Zhang et al., 2019, Chen et al., 2018] for Problem 1 is to obtain initial pose estimates by tracking the pose relatively for the recorded image sequence or with help of GNSS measurements. Subsequently, local map consistency of revisited places with available image overlaps (loop closures) is ensured through refinement of the estimated image poses, considering correspondences of neighboring images in a nonlinear least-squares optimization process. A variant of such an approach, which we use for the scanning process of our own database, is introduced in Section 3.3.3.

Problems 2 and 3 are solved in conjunction as absolute localization requires the map data structure as input. Here, the state-of-the-art methods [Kozak and Alban, 2016, Zhang et al., 2019, Chen et al., 2018] are feature-based. For map creation, they extract characteristic image patches (local features) from the reference images. Subsequently, for localization, they determine correspondences

with the query image features. A more detailed introduction of the current state of the art for ground texture based mapping and localization approaches is given in Chapter 2.

1.3 Open Research Questions

While existing ground texture based localization methods are able to localize reliably and accurately in many cases [Kozak and Alban, 2016, Zhang et al., 2019, Chen et al., 2018], they have been constructed without extensive consideration of possible alternatives. For example, the employed feature extraction pipelines are selected without any reasoning [Chen et al., 2018], or based on simple replacement experiments [Kozak and Alban, 2016, Zhang et al., 2019], that consider only a small selection of available techniques, and that do not consider parametrization of the employed techniques, which are by default not optimized for ground images. In this respect, it is also an open question to what extent parameter adjustments can improve the localization performance and how to efficiently find suitable parameter settings.

Furthermore, performance of the state-of-the-art localization methods is reported without comparison to each other, leaving the question of which method works best in which situation. In particular, there are scenarios of interest that have not been considered systemically during the performance evaluation of ground texture based localization methods, e. g. how does localization performance behave if the current position is known within certain bounds or how are natural changes of the ground that occur over time, or due to weather changes, affecting the localization capabilities.

A remaining challenge for the existing localization methods is their real-time applicability, as they report computation times of about 100 ms [Chen et al., 2018, Kozak and Alban, 2016] to 245 ms [Zhang et al., 2017], which is dominated by the required time for feature extraction and the subsequent process of determining feature correspondences (matching) [Zhang et al., 2017]. This leaves the question of possible alternative cost-effective solutions.

This dissertation aims to address these open research questions.

1.4 Contributions

In the course of my doctoral studies, I was author of several scientific publications on the introduced subject. Their content constitutes the essential part of this dissertation. The following list presents the publications in chronological order of their publication date. It also states explicitly what my part, respectively that of the co-authors, in these works was. Furthermore, for all listed contributions, I did the essential part of the write-up, and I am the main author, only in the case of [Radhakrishnan et al., 2021], I share this position with my co-author Raaghav Radhakrishnan.

- Schmid, Simon, and Mester [2019] (peer-reviewed). This publication provides the first extensive evaluation of available methods for the extraction of local visual features for their suitability to the task of finding correspondences between pairs of ground images. Here, we considered both independently recorded images and synthetically transformed ones, and we examined image pairs with similar orientations recorded in direct succession, as well as image pairs that were recorded completely independent of each other which in this case often have large orientation differences. The contribution of this work is the identification of the most suited pipelines for feature extraction on ground images, and the determination of appropriate parametrization for them.

  My part in this work was the literature analysis for the most promising existing feature extraction methods, and for existing surveys of local visual features; the conceptualization and design of the experimental framework, as well as the implementation, execution, and analysis of the experiments. The co-authors engaged in discussions with me about the research procedure and the results obtained. Also, they supported me with helpful comments on the manuscript.

- Schmid, Simon, and Mester [2020a] (peer-reviewed). In this work, we introduce an extensive evaluation framework for ground texture based localization methods and we use it to perform the first systematic comparison of existing approaches. Also, we present a novel ground texture based localization method that builds on the state-of-the-art ground texture based localization method Micro-GPS, developed by Zhang et al. [2019]. Our method is self-contained, which means that it could be used as the exclusive source of positioning information for an autonomous agent. This is because, it is well suited for both localization with and without available prior. The method is based on the employment of compact

binary descriptors and a novel matching technique for visual features that we call *identity matching*, which matches only those features that receive bit-identical feature descriptors. In our experiments, we observe higher localization success rates as that of the state-of-the-art methods. Also, the method to be particularly efficient to compute if a prior is available.

My part in this work was the literature review and re-implementation of existing ground texture based localization approaches; the experimental evaluation; the implementation and analysis of proper design decisions for our proposed method, as well as its parametrization. The co-authors engaged in discussions with me about the research procedure, the examined concepts, and the results obtained. Also, they supported me with helpful comments on the manuscript.

- Schmid, Simon, and Mester [2020b] (peer-reviewed). In this follow-up paper, we are examining the idea of feature-based ground texture based localization without the employment of proper keypoint detection, i. e. local visual features are computed for arbitrarily sampled image positions. In our results, we observe that all three evaluated state-of-the-art ground texture based localization methods, when using this approach, reach similar localization success rates than with their regular methods for keypoint detection. However, our method has the greatest benefit in terms of the resulting localization runtimes. This examination suggests that it is not necessary to recognize distinctive features of the ground for successful localization. Instead, the consideration of arbitrary ground regions seems to be sufficient for this task. Based on the keypoint sampling approach, this work contributes a real-time capable CPU-only ground texture based localization method that presents high localization success rates on the examined ground texture types.

  My part in this work was the literature review for related work; the development of the proposed method, including the evaluation of various design decisions, extensions, and the parametrization; the introduction of suitable performance metrics to be analyzed in the experimental evaluation; as well as the implementation and execution of the experimental framework and analysis of its results. The co-authors engaged in discussions with me about the research procedure, the examined concepts, and the results obtained. Also, they supported me with helpful comments on the manuscript.

- Schmid, Simon, and Mester [2021]. This paper presents a model-based approach to efficient estimation of the global localization performance

of ground texture based localization methods. We derive a predictive model based on stochastic means that requires only a small sample set of ground images of an application area to approximate the expected localization performance. The model is applicable to our own localization method, as well as to Micro-GPS [Zhang et al., 2019], and it allows to make appropriate decisions about any of their performance influencing parameters. Accordingly, we are able to use the model in an automatic parameter optimization framework, which we observe to be able to find suitable texture-specific parameters in only a few hours of time, which is in contrast to the week-long manual parametrization process we did in previous works.

My part in this work was the literature review of existing related work; the conceptualization and derivation of the prediction model; the implementation of the prediction model for practical use and the optimization framework; as well as the experimental evaluation and the analysis of results. The co-authors engaged in discussions with me about the research procedure and the results obtained. Also, they supported me with helpful comments on the manuscript.

- Radhakrishnan, Schmid, Scholz, and Schmidt-Thieme [2021]. This paper presents a novel approach to the task of image retrieval of ground images, i. e. given a query image, it finds the reference images that overlap with it. For this purpose, we use a deep metric learning approach based on a Siamese Convolutional Neural Network (CNN) and an objective function similar to that of Sánchez-Belenguer et al. [2020]. We show that this approach has significantly better recall performance than the current state of the art for this task based on the Bag of Words (BoW) technique. Also, we are able to increase global localization performance of our ground texture based localization method slightly compared to the case in which we do not use image retrieval and significantly in the case in which we are using BoW ground image retrieval.

  My part in this work was the proposal to build on the overlap loss proposed by Sánchez-Belenguer et al. [2020] and to transfer it to the use case of ground image retrieval; the implementation of the baseline method based on the BoW technique, as well as the baselines methods based on the work of [Revaud et al., 2019] and [Gordo et al., 2017]; the evaluation of the retrieval results for localization with the previously proposed ground texture based localization method; as well as the general supervision of the project. Raaghav Radhakrishnan contributed the literature review of

related work; the implementation and evaluation of the proposed deep metric learning model, as well as its variants, and of the baselines *Random*, ResNet [He et al., 2016], and DenseNet [Huang et al., 2017]; the parametrization of the BoW baseline; as well as the creation of visualizations for the manuscript. Together, Raaghav Radhakrishnan and I, developed the evaluation setup, including the evaluated performance metrics. The remaining co-authors engaged in discussions with us about the research procedure and the results obtained. Also, they supported me with helpful comments on the manuscript.

- Schmid, Simon, Radhakrishnan, Frintrop, and Mester [2022] (peer- reviewed). This paper presents the *HD Ground Database* and the setup of our own robot with a downward-facing camera, that we used to record the database (see Figure 1.3). To date, the HD Ground Database is the largest collection of ground images that allows to evaluate localization performance of methods that rely solely on ground images in varying real-world indoor and outdoor application areas. Also, in contrast to existing databases, our robot setup allows us to record higher-quality images with particularly small amounts of motion blur, due to a short exposure time of about 0.1 ms. We use the database for a re-evaluation of the current state of the art of ground texture based localization methods, and we provide the first systematic examination of how localization performance degrades when the ground exhibits progressive changes with the time between map creation and localization increasing. Additionally, with this database it is the first time we have had the opportunity to examine a teach-and-repeat scenario, where the robot is supposed to follow a previously taught path autonomously in either direction.

  My part in this work was the literature review of related work; the conceptualization and execution of the data recording strategy; the design of the recording setup, including our own recording platform, which was constructed as part of a student project under my supervision; preparation and processing of the recorded images, including the generation of ground truth poses for reference and query images; as well as the implementation of the experimental setup and the analysis of its results. Raaghav Radhakrishnan contributed the application of our previously developed image retrieval approach on the database. The remaining co-authors engaged in discussions with me about the research procedure, the results obtained, and helped with the construction of the recording platform. Also, they supported me with helpful comments on the manuscript.

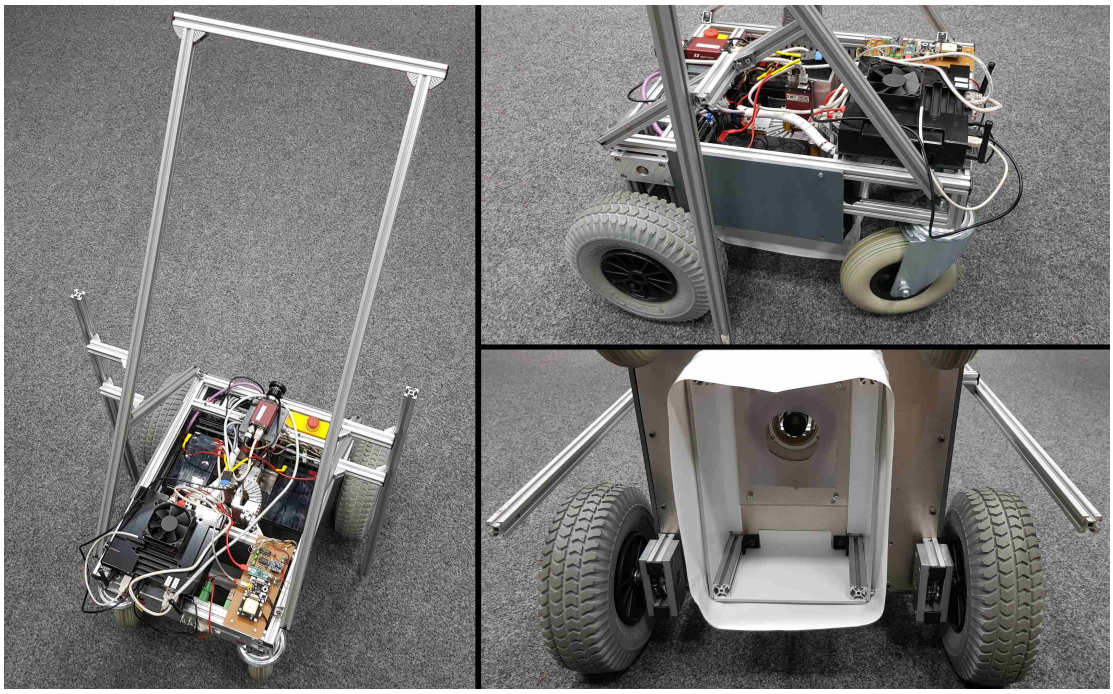


Figure 1.3: We modified an RT3-2 VolksBot [Surmann et al., 2008] for data recording. The image on the left shows the setup from above, the top right one shows it from the side, and the lower right one from underneath. Aluminium profiles are added as handle bar and for visual guidance during manual line following. An AVT Manta G-235C camera is used together with synchronized pulsed LED lighting to record images of the ground below the robot. White plastic film prevents sunlight from falling directly on the recorded area.

1.5 Structure

This document is structured as follows.

Chapter 2 presents the technical background and the related work that is common to multiple of the following chapters. Topic-specific background information, on the other hand, is provided in the respective chapters.

The following six chapters present novel contributions to the field of ground texture based localization. While there are a few cross references between the chapters, they can be read independently of each other. The chapters are roughly aligned with the aforementioned publications.

Chapter 3 presents the ground image databases of Zhang et al. [2019], which we call *Micro-GPS databases* in the following, and which were previously the only databases that could be used to examine ground texture based localization performance. Additionally, the chapter presents the HD Ground Database and our own robot that we used to record the database. These are contributions from [Schmid et al., 2022].

Subsequently, Chapter 4 presents contents of [Schmid et al., 2019], where we presented a survey about the suitability of existing feature extraction strategies for the task of finding correspondences between ground image pairs.

Chapter 5 introduces the *GTBL Method* and evaluates on a Micro-GPS database together with the existing state of the art for localization methods using ground images, which are contributions from [Schmid et al., 2020a]. In addition to that, we present the re-evaluation of the same methods from [Schmid et al., 2022] on the HD Ground Database.

The content of [Schmid et al., 2021], in which we derive a prediction model for global localization performance, is presented in Chapter 6.

Then, Chapter 7 presents our deep metric learning approach to the retrieval of ground images from [Radhakrishnan et al., 2021]. Here, again, we present additional evaluation to that from the original publication from [Schmid et al., 2022] on the HD Ground Database.

Similarly, Chapter 8 provides the examination of our keypoint sampling strategy for ground texture based localization from [Schmid et al., 2020b], and it is supplemented with additional evaluations on the HD Ground Database from [Schmid et al., 2022].

Finally, in Chapter 9, we assess the research questions of Section 1.3 and we conclude with an outlook for possible future work.

2 Background

Contents of this chapter were partially published in [Schmid, Simon, and Mester, 2019], [Schmid, Simon, and Mester, 2020a], [Schmid, Simon, and Mester, 2020b], [Schmid, Simon, and Mester, 2021], and [Schmid, Simon, Radhakrishnan, Frintrop, and Mester, 2022].

2.1 Local Image Features

In image processing, *features* are used to describe characteristic properties of images [Pratt, 2007]. A feature is called *local* if it is only influenced by a spatially restricted image region [Tuytelaars and Mikolajczyk, 2008], which we refer to as *image patch*. In the following, we will use the term *feature* for local image features. The image coordinates of the image patch are specified by the feature's *keypoint* (in the literature also referred to as interest point) and its content is represented by the feature *descriptor*. In addition, we introduce the term *keypoint object*, which refers to the complete spatial description of the image patch. Typically, this includes the keypoint, as well as a size (sometimes called *scale*) and an orientation.

Keypoint objects are extracted usually by a *keypoint detector* that searches for characteristic patterns like edges, corners as in [Shi and Tomasi, 1994, Rosten and Drummond, 2006, Mair et al., 2010], or blobs that stand out to their surrounding as in [Lowe, 2004, Bay et al., 2006, Agrawal et al., 2008], e. g. extrema in the Difference-of-Gaussian (DoG) pyramid.

Features are useful to relocate the same physical location or to identify an object that is visible in one image in another image taken with a different camera pose. However, the same physical location or object will have varying appearances: from varying distances the corresponding features are represented by varying amounts of pixels in the image, at different recording times the illumination might differ, and corresponding features appear at different image locations and with varying orientation.

This poses the challenge of identifying the same features under varying appearance transformations. A feature detector or feature description method can be *robust* to these variances, i. e. relatively small deformations still lead to the similar results [Tuytelaars and Mikolajczyk, 2008]. In some cases, like for the image patch rotation, the method can even be designed to be *invariant* to changes of the property, i. e. the method is provably unaffected to a certain type of transformation [Tuytelaars and Mikolajczyk, 2008].

Dependent on the use case, in order to perform reliable relocation of features, it is desirable that features are robust to one or more of these transformations. Robustness to scale, for example, can be achieved by searching for features on multiple image scales that are computed through smoothing and subsampling. From a signal processing stand point smoothing is equivalent to removing high frequency information applying a low-pass filter. Subsampling creates image versions of different sizes and allows to use the same descriptor pattern, i. e. the same arrangement of sampled pixels around the keypoints, to detect features of varying scales. Illumination variances, on the other hand, can be reduced through normalization of pixel intensities or their derivatives.

While it seems desirable to be invariant to as many changes of appearance as possible, Tuytelaars and Mikolajczyk [2008] explain that "[...] an increased level of invariance typically leads to a reduced distinctiveness, as some of the image measurements are used to lift the degrees of freedom of the transformation." Accordingly, the employed methods for feature extraction should be selected carefully dependent on the use case. In the context of ground texture based localization, we might assume a constant distance between the downward-facing camera and the observed ground. A close to constant feature scale follows from that. Also, if we employ our own lighting of the ground, illumination variances are kept at a low level. The requirement for robustness against rotation depends on the exact application. In a teach-and-repeat scenario, in which a robot is asked to move along a pre-defined path, which it is manually driven along during the teach-phase, the moving direction of the robot is known quite precisely. Accordingly, we can do without orientation invariance in this case. Similarly for incremental localization updates, where the approximate robot pose is known with good accuracy.

Once features are available, in order to find correspondences in different images, a feature matching algorithm is employed [Szeliski, 2010]. Typically, it is unlikely to obtain the same descriptor for corresponding physical locations, because feature description algorithms are not able to compensate for all variances of the feature appearance. Therefore, features with similar (rather than identical)

feature descriptors are matched with each other. The conventional approach to this is to define a metric to measure the distance between descriptors, and then most similar descriptors, i. e. nearest neighbors in the descriptor space, can be identified as correspondence candidates. In a naive brute-force approach, distances between each pair of descriptors of two images are computed. However, in time-critical applications an approximate solution can be used ([Yan et al., 2015]). In such a case, only Approximate Nearest Neighbors (ANNs) are available for feature descriptor matching.

In the following, we present the methods for keypoint detection (Section 2.1.1), and feature description (Section 2.1.2) relevant to this dissertation.

2.1.1 Keypoint Detection

Keypoint detection is concerned with the task of finding characteristic visual phenomena that are likely to be relocated in other images containing the corresponding physical location.

Keypoint detection approaches can be split into corner detectors and scale-space detectors [Agrawal et al., 2008]. Corners mark suitable keypoints as they tend to be robust to view changes. The **Harris** detector [Harris and Stephens, 1988] and Good Features To Track (**GFTT**) [Shi and Tomasi, 1994] approximate the second derivative of the sum-of-squared-differences with respect to the shift of a circular image patch to detect edges and corners. A corner is found if both principle curvatures of the local auto-correlation function are high. An edge is found if one curvature is high and the other is low. **FAST** [Rosten and Drummond, 2006] compares intensities of center pixels with their surrounding pixels on a circle. A corner is detected if the circle contains a contiguous sequence of pixels with significantly larger or lower intensity values. If this condition can no longer be fulfilled it can be rejected early. To do this, a decision tree defines the order of comparisons. Mair et al. [2010] adapt this concept for **AGAST**. Instead of using a single decision tree, they switch between multiple ones according to observed local image characteristics.

Scale-space detectors exploit image scale pyramids to find scale invariant keypoints. Mikolajczyk and Schmid [2002] extended with **Harris Laplace** the Harris corner detector to search for corners in multiple scales using a Gaussian scale-space. **SIFT** [Lowe, 2004] detects blobs using a DoG pyramid as local minima and maxima of the intensity values in scale and space. Candidates located on edges or with low contrast are suppressed. Orientation is determined by the dominant local intensity gradients. **SURF** [Bay et al., 2006] and

CenSurE [Agrawal et al., 2008] approximate the DoG with bi-level Laplacian of Gaussian like Difference-of-Boxes (DoB) or Difference-of-Octagons (DoO), which can be computed efficiently using integral images. While SIFT and SURF find keypoints using the Hessian measure, CenSurE relies on the Harris corner response. **BRISK** [Leutenegger et al., 2011] and **ORB** [Rublee et al., 2011], on the other hand, use efficient corner detectors like FAST on a scale pyramid to identify repeatable keypoint objects in scale-space. Alcantarilla et al. [2012, 2013] argue that Gaussian scale-space pyramids and its approximations do not only remove noise, but interesting image details as well. Therefore, they propose the method **AKAZE** that finds keypoints as maxima of the Hessian in non-linear scale-space.

The **MSER** [Matas et al., 2004] method follows a similar approach to the watershed segmentation algorithm [Soille and Vincent, 1991]. The image is thresholded by an increasing illumination value. Regions with illumination values below the threshold emerge and grow during this process. Keypoint objects are identified as regions at their point of slowest growth. In **MSD** [Tombari and Di Stefano, 2014] image regions that differ from their surrounding in a large neighborhood are considered as keypoint objects.

2.1.2 Feature Description

Given an image patch defined by a keypoint object, feature description methods are used to produce a characteristic compact representation of its visual content. An ideal feature description method should be robust to small variances of the image patch and its content, and its generated descriptors should be specific (optimally they would be unique) to the underlying physical location, while they should contain as little redundant information as possible, i. e. be represented with as few bits as possible, for efficient further processing.

Historically, feature descriptors are real-valued. **SIFT** [Lowe, 2004] describes keypoint objects using a histogram of gradient directions. Similarly, **DAISY** [Tola et al., 2010] also uses quantized orientation histograms. But, its histogram bins are distributed radially around the keypoint and smoothed increasingly with the distance to the keypoint. **SURF** [Bay et al., 2006] relies on Haar-Wavelet responses that are efficient to compute using integral images.

More recently, research started to focus on compact binary feature descriptors [Pietikäinen et al., 2011]. Most of them construct descriptors as concatenated results of pairwise intensity comparisons. **BRIEF** [Calonder et al., 2010] compares randomly paired pixels from a smoothed image patch. **ORB** [Rublee

et al., 2011] uses the same approach as BRIEF, but it employs a training algorithm to determine the set of pixel comparisons and rotates this pattern according to the keypoint object orientation. **BRISK** [Leutenegger et al., 2011] samples short- and long-distance pixel pairs around the keypoint. While short-distance pairs are evaluated for the descriptor, long-distance pairs are used to determine an orientation. A similar approach is employed by **FREAK** [Alahi et al., 2012], but suitable intensity comparisons are found in a training process. In **LATCH** [Levi and Hassner, 2016] triplets of image patches are compared to each other instead of pixel pairs to increase robustness. **AKAZE** [Alcantarilla et al., 2013] performs pairwise comparisons of first-order gradients.

Most recently, deep learning approaches for feature extraction are developed. **LIFT** [Yi et al., 2016] is a state-of-the-art method that provides solutions for keypoint detection, orientation estimation, and feature description. The proposed network is trained in Siamese fashion with features from a Structure-from-Motion (SfM) algorithm, using images from photo-tourism datasets containing many views of the same landmarks. The architecture can be trained end-to-end, solving the whole feature extraction task in a single forward-pass. However, in practice they train the network separately for the three tasks. Training samples consists of four image patches, two corresponding ones for which the network learns to produce similar output and two other patches that should result in distinctively different network outputs.

2.2 Feature-Based Localization

A common approach to visual self-localization of autonomous agents is to identify correspondences between mapped reference images and the query image, representing the current view of the agent.

The set of considered reference images might contain all available reference images of the entire application area for which we know their respective poses in a map coordinate system (global map-based localization), or it might contain just the reference images in the local vicinity of our current estimate of the robot pose (local map-based localization), or it could be just the previous recording the robot acquired (incremental localization).

The required correspondences can be found with photometric approaches [Kelly et al., 2007], that compare image patches based on a function of image intensity values, e. g. normalized cross correlation, or with feature-based approaches, that propose well-matching features as correspondences [Zhang et al., 2019,

Fang et al., 2009, Nagai and Watanabe, 2015, Kozak and Alban, 2016, Chen et al., 2018]. Current state-of-the-art approaches to ground texture based localization rely on feature-based localization [Kozak and Alban, 2016, Chen et al., 2018, Zhang et al., 2019], which is why in the following we will focus on this approach.

Feature-based localization can be divided into 5 subtasks:

1. *keypoint detection*, finding the same keypoint objects in query and reference images from different viewpoints and under varying photometric conditions like illumination, noise, and blur;
2. *keypoint selection*, selecting a subset of keypoint objects for further processing;
3. *feature description*, computing descriptors that robustly take similar values for corresponding keypoint objects, and distinctively different values for non-corresponding ones;
4. *feature matching*, proposing feature correspondences between the query and the reference images;
5. *pose estimation*, estimating the query image pose with respect to the reference image poses, based on the proposed correspondences.

State-of-the-art ground texture based localization methods are built according to this scheme, employing various detector-descriptor pairings, e. g. CenSurE-ORB [Kozak and Alban, 2016], SURF-SURF [Chen et al., 2018], or SIFT-SIFT [Zhang et al., 2019].

For feature selection, most keypoint detectors provide a keypoint score, e. g. representing the response strength to the employed detection criterion [Lowe, 2004], that can be used to either consider only the top-n features, or to consider only those features with keypoint score above some threshold. Alternatively, possibly to increase the variety of the kept feature set, a subset of the detected features can be sampled randomly, e. g. [Zhang et al., 2019]. Features are typically matched using costly brute-force pair-wise matching as in [Kozak and Alban, 2016, Chen et al., 2018] or ANN matching [Zhang et al., 2019].

Only a subset of the set of proposed matches $\mathcal{M}$ can be considered to be correct: the set of *inlier* matches $\mathcal{I}$. The remaining matches are considered to be incorrect, they form the set of *outlier* matches $\mathcal{O}$ (with $\mathcal{M} = \mathcal{I} \cup \mathcal{O}$). Accordingly, the subsequent pose estimation step has to deal with this situation in which potentially a significant share of proposed correspondences are incorrect. For this purpose, all state-of-the-art approaches employ pose estimation in RANSAC fashion [Kozak and Alban, 2016, Chen et al., 2018, Zhang et al., 2019].

RAndom SAmple Consensus (RANSAC) [Fischler and Bolles, 1987] can be used to estimate the optimal parameters for a model. For this purpose, RANSAC determines a set of inliers from a set of data points, which might also contain outliers. In the scenario of a dataset that contains both inliers and outliers, RANSAC can be superior to other model fitting methods like least squares optimization. The least squares method does not differentiate between inliers and outliers, and therefore fits the model that is the best explanation to the measurements of all data points. RANSAC, on the other hand, determines the best fitting models of multiple randomly sampled data points. Internally, RANSAC uses a voting scheme to find the best fitting model. Each data point votes for models that explain its occurrence. The assumption is that outlier votes are not consistent, while all inliers can be explained with the same model. Typically, each randomly sampled set of data points only contains as many data points as are required to fit parameters of a model. Then, it is examined how many of the remaining data points can be explained with the obtained model (as well as a certain amount of noise deviation), these are considered inliers, all others are considered outliers. The algorithm repeats to determine the set of inliers (consensus set) for randomly sampled data subsets until a model with a sufficiently large consensus set is found. Afterwards, the final set of inliers can be used jointly to determine the output model.

In the case of ground texture based localization, with an assumed constant distance between ground and camera, the localization problem is a 2D problem in which the camera pose can be described by a standard Euclidean transformation with three variables: the x- and y-coordinates and an orientation angle. Accordingly, a single feature correspondence with available keypoint object orientation information, or two feature correspondences without available keypoint object orientation information, are sufficient to create a model, i. e. a query image pose estimate. Once the RANSAC procedure terminates successfully, the final query image pose estimate can be computed considering all *RANSAC inliers*, i. e. all feature correspondences of the final consensus set.

2.3 Voting Procedure for Spatial Verification of Feature Matches

The aforementioned RANSAC procedure for pose estimation becomes slow to compute if the share of inliers is small compared to the number of outliers. This is because the chance of having a sample set of only correct matches becomes

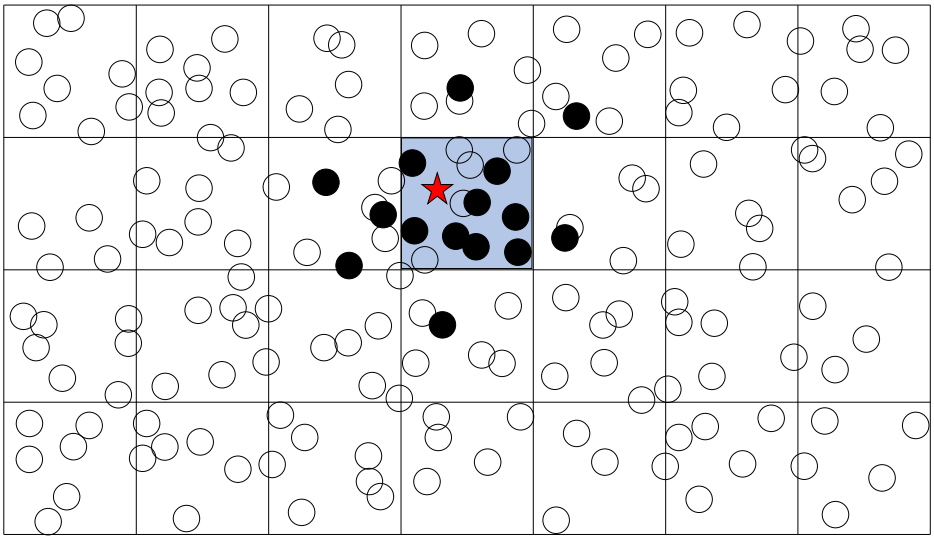


Figure 2.1: Illustration of the voting procedure of Zhang et al. [2019] for spatial verification of feature matches. The entire rectangle represents a map of the (rather small) application area. A 2D histogram splits the map into equally sized voting cells. Every proposed correspondence from the feature matching step votes for the query image position (circles). Some matches, the inliers, represent correct feature correspondences (solid circles). They provide the required information to find the correct query image pose estimation. Inlier votes concentrate close to the true query image position (red star), while outlier votes (transparent circles) are distributed randomly. Only the matches voting for the cell that received most votes (blue), which we call *voting peak*, are used for the subsequent pose estimation step. This figure is adapted from [Schmid et al., 2021].

small. Accordingly, many incorrect models, i. e. query pose estimates, will be considered. Depending on the termination criterion, the procedure might even stop before the inlier consensus set is found.

Global map-based localization, which considers features of all available reference images for feature matching, can lead to such an unfavorable situation [Zhang et al., 2019]. In such cases, prior rejection of outliers is desirable.

One approach to outlier rejection is the spatial verification of feature matches, using Hough voting approaches [Avrithis and Tolias, 2014, Schönberger et al., 2017, Zeisl et al., 2015]. The idea to use a voting procedure for spatial verification of feature matches from ground images was proposed by Zhang et al. [2019]. The procedure is illustrated in Figure 2.1. Their proposed technique exploits the fact that every match of ground features represents a query image pose estimate. A match m is a pair of features, one from the set of query image features $\mathcal{F}_\mathrm{q}$ and one from the set of reference features $\mathcal{F}_\mathcal{R}$. Given the match $m = (f_q \in \mathcal{F}_\mathrm{q}, f_r \in \mathcal{F}_\mathcal{R})$, we can derive an estimate of the query image pose

$[\mathbf{R}_{\text{est}}|\boldsymbol{t}_{\text{est}}]_{\text{q}}^{\text{M}}$ based on three transformations: (a) the transformation $[\mathbf{R}|\boldsymbol{t}]_{\text{q}}^{f_q}$ that maps from the query image pose to the pose of f_q, (b) the transformation $[\mathbf{R}|\boldsymbol{t}]_{f_r}^{\text{M}}$ that maps from the reference feature pose to the map coordinate system, and (c) the transformation $[\mathbf{R}|\boldsymbol{t}]_{f_q}^{f_r}$ that maps from the query feature pose to the reference feature pose, which, due to the assumed (correct) correspondence of f_q and f_r, is estimated to be the identity. The query image pose is estimated as

$$[\mathbf{R}_{\text{est}}|\boldsymbol{t}_{\text{est}}]_{\text{q}}^{\text{M}} = [\mathbf{R}|\boldsymbol{t}]_{f_r}^{\text{M}}[\mathbf{R}|\boldsymbol{t}]_{f_q}^{f_r}[\mathbf{R}|\boldsymbol{t}]_{\text{q}}^{f_q} = [\mathbf{R}|\boldsymbol{t}]_{f_r}^{\text{M}}[\mathbf{R}|\boldsymbol{t}]_{\text{q}}^{f_q}. \tag{2.1}$$

In order to reject outliers, the voting procedure proceeds as follows. Every proposed match votes for the position of the query image in the map coordinate system, using the translation component of their corresponding pose estimate. The orientation component is ignored, as it tends to be inaccurate. Votes cast into a similar area are summarized using a 2D histogram, i. e. a grid of equally sized voting cells. The overall grid size corresponds to that of the mapped environment. We call the voting cell with most votes the *voting peak*. Only the matches contributing to the voting peak will be considered during the subsequent pose estimation step, as we expect some of them to be inliers. Even though inliers, in particular due to inaccuracies in the keypoint object orientations, are not necessarily voting for positions precisely at the true query image position, we expect them to concentrate close to it, while outlier votes are scattered randomly over the voting histogram. Due to the random distribution of outliers, which we examine in more detail in Chapter 6, there will be regions on the voting histogram sparsely populated with outlier votes and there will be regions densely populated with outlier votes. If the peak caused by the inliers is larger than any peak caused by outlier votes, the approach succeeds in its task of outlier rejection; otherwise, the actual inliers will be rejected as outliers.

2.4 Approaches to Ground Texture Based Localization

Ground texture based localization builds on the observation that ground image patches can be used as fingerprint-like identifiers [Zhang et al., 2019]. For the applications considered in this dissertation, e. g. robots moving autonomously through a warehouse or an apartment, it is reasonable to assume that the ground is locally flat and therefore that the distance to the ground is known. Accordingly, with a downward-facing camera, pose estimation is reduced to

determine two coordinates for the position and one orientation angle. This corresponds to a standard Euclidean transformation of rotation and translation in two dimensions.

Ground texture based localization can be performed with *appearance-based approaches* [Aqel et al., 2016, Zaman, 2007, Kelly et al., 2007, Nagai and Watanabe, 2015], e. g. using normalized cross-correlation to find reoccurring image patches, and with *feature-based approaches* that find feature correspondences [Swank, 2012, Nakashima et al., 2019, Kozak and Alban, 2016, Zhang et al., 2019, Chen et al., 2018]. Furthermore, localization methods can be divided into approaches for map-based *absolute localization* with or without available prior pose estimate, and approaches for incremental *relative localization* to estimate the pose of the current camera image in respect to the previous one [Desouza and Kak, 2002].

2.4.1 Absolute Localization

Absolute localization methods determine the query image pose in respect to a given map, i. e. a set of reference images with known poses in a common map coordinate system.

Two cases of absolute localization are to be distinguished: a) global localization, and b) local localization. The global localization task requires to consider all available mapped reference images, as there is no knowledge about the current robots whereabouts. Global localization solves the kidnapped robot problem, which occurs whenever an agent is not aware of how it got to its current place. If, however, a prior is available, it is sufficient to consider only those reference images that are spatially close to the estimated query image pose.

In the following, Section 2.4.1.1 and Section 2.4.1.2 present approaches to global absolute localization, respectively local absolute localization.

2.4.1.1 Global localization approaches

Micro-GPS is a localization pipeline proposed by Zhang et al. [2019]. They rely on SIFT [Lowe, 2004] for feature extraction, and construct an efficient Approximate Nearest Neighbor (ANN) search structure for feature matching. Here, they assign the extracted reference image features into 10 groups based on their keypoint object scale. An ANN search index is constructed for each group. This exploits the fact that the scale of corresponding features remains essentially constant for images of a downward-facing camera with stable height, because,

during the feature matching step, it is sufficient to search for correspondences for a query image feature in the group of reference features with similar scale. Per reference image 50 randomly sampled features are inserted into their respective search indexes. During localization, all query image features are used for feature matching. Each feature from the query image is paired with its ANN. The previously introduced voting procedure is employed for outlier rejection (see Section 2.3). Finally, only the correspondences contributing to the voting peak are used to estimate the camera pose in RANSAC fashion.

Chen et al. [2018] developed *StreetMap*, which is able to make use of a localization prior, but does not require one. While there is also a version of StreetMap specifically for tiled ground textures, we only consider the feature-based variant. If no prior is available, they use BoW image retrieval [Galvez-López and Tardos, 2012] to find similar reference images to the query image. For this purpose, BoW representations of the images are computed using the SURF [Bay et al., 2006] feature descriptors extracted from them. After retrieval of the reference images with most similar BoW representation, their features are matched to the features of the query image. For each feature of the query image, they search for its nearest neighbor from the reference images and subsequently filter these matches with the *ratio test constraint* [Lowe, 2004], which requires that the most similar reference descriptor is significantly closer to the query descriptor than the second most similar one. The Euclidean transformation of the camera pose is finally estimated in a RANSAC procedure.

The work of Macias-Sola et al. [2021] builds on concepts of Micro-GPS and the localization method that we introduced in [Schmid et al., 2020a]. As in our method, which will be described in detail in Chapter 5, the authors use SIFT [Lowe, 2004] to extract keypoint objects and LATCH [Levi and Hassner, 2016] to determine compact feature descriptors (compact in this case means 16 bits per descriptor), and they also propose only those features as matches that have identical descriptors. In order to retrieve a mapped reference image that can be used to match its features with that of the query image, the authors employ a BoW-like image retrieval approach, using the histogram of feature descriptor values of an image as image descriptor. However, the authors notice that their method for image retrieval is not robust. This is, even though they evaluate their approach on comparatively simple application areas, consisting of short straight paths.

2.4.1.2 Local localization approaches

Kelly et al. [2007], Kelly [2000] developed a photometric localization approach using normalized cross correlation for template matching to find corresponding image patches between query and reference images. They construct a ground map of statistically normalized pixel intensity values using image stitching. During localization, the output of a Kalman filter is used as a localization prior. Peaks of a texture score function, which depends on the local intensity gradient of the pixels, are used to define up to 16 image patches for template matching. The difference between predicted and observed positions of these image patches is combined into a pose update.

The localization pipeline of Fang et al. [2009, 2007] relies on the Iterative Closest Point (ICP) algorithm to align reference images during mapping and to register query images for localization. The point clouds needed for this purpose are built using corner and edge features extracted from the images. For the final pose estimate, the results of a robust ICP variant are fused with odometry information in an extended Kalman filter.

Nagai and Watanabe [2015], Nagai [2007] propose a method that avoids the need for a globally consistent map. Instead, they construct a sparse spatial map of images. Whenever the autonomous system approaches a reference image stored in the map, correspondences between query and reference image are used to correct for the drift that accumulated since the last absolute localization step. Image transformations are estimated through minimization of the reprojection error, which is measured as cross-correlation of intensity values.

Kozak and Alban [2016] developed *Ranger*, a method that enables localization at high vehicle speeds of up to 120 km/h. Ranger computes ORB [Rublee et al., 2011], respectively rotated BRIEF [Calonder et al., 2010], feature descriptors for CenSurE [Agrawal et al., 2008] keypoint objects. Ranger iteratively considers the spatially closest reference image to the given prior to match its features with that of the query image, using brute-force nearest neighbor matching. A *cross-check* is performed to reject incorrect matches. This means that in order for the reference image feature $f_r \in \mathcal{F}_\mathcal{R}$ and the query image feature $f_q \in \mathcal{F}_\mathrm{q}$ to be considered a match, f_r needs to be the nearest neighbor of f_q among all reference features $\mathcal{F}_\mathcal{R}$ and f_q needs to be the nearest neighbor of f_r among all query features $\mathcal{F}_\mathrm{q}$. The remaining correspondences are used to perform RANSAC-based pose estimation. If at least 25 correspondences are consistent with the obtained pose estimate, i. e. if the consensus set has a size of at least 25, the pose estimate is used as the final pose estimate output. Otherwise, the

procedure is repeated with the next closest reference image (or localization is aborted due to timeout).

As previously mentioned, StreetMap can also be used for local localization, making use of a given prior [Chen et al., 2018]. In this case, instead of selecting reference images based on BoW similarity, the images with shortest spatial distance to the prior are taken into consideration.

2.4.2 Relative Localization

Relative, respectively incremental, localization methods determine the pose of the most recent image recording with respect to the previously recorded images (e. g. [Nourani-Vatani et al., 2009])

Appearance-based approaches like that of Zaman [2007] and Gilles and Ibrahimpasic [2021] can perform relative localization by estimating transformations between ground images directly based on the observed appearance changes. For example, in the case of Gilles and Ibrahimpasic [2021], using a deep neural network that is trained in an unsupervised manner for image registration.

Feature-based approaches, on the other hand, track local visual features in consecutively recorded images. For example, Nakashima et al. [2019] proposed a solution based on AKAZE [Alcantarilla et al., 2013] features. They propose to search for correspondences of a given feature from a previously recorded image by predicting its position in the current image, which avoids to match the feature with all features of the current image, improving robustness and efficiency of the method.

Furthermore, methods for local absolute localization like Ranger [Kozak and Alban, 2016] and StreetMap [Chen et al., 2018], can also be employed for relative localization, using only the previous query image as reference image.

Methods for incremental localization, that estimate the vehicle pose relative to a previous pose, accumulate drift and are therefore usually accompanied by an error correction mechanism, e. g. an absolute localization method.

3 Databases for Ground Texture Based Localization

Contents of this chapter were partially published in [Schmid, Simon, Radhakrishnan, Frintrop, and Mester, 2022].

An essential basis for the development and evaluation of methods and techniques for ground texture based localization is the availability of data in form of ground images that have been recorded in a suitable and structured manner. This is to solve the Problem 1 of ground texture based localization: the initial scanning of the application area. For this purpose, we contribute the HD Ground Database, a comprehensive database of ground images, obtained using a downward facing camera, which is supported by our own lighting of the ground and which is shielded from external light sources. The database enables the examination of localization under varying conditions, such as clean versus dirty, and dry versus wet ground. Also, in comparison to existing ground image databases, the HD Ground Database provides larger area coverage, has a greater variety of textures, higher resolution images with less motion blur, and image sequences that allow to evaluate a teach-and-repeat scenario in which the robot is supposed to follow a previously learned path. Most importantly, the HD Ground Database provides the novel opportunity to examine degradation of localization performance that occurs for increasing intervals between the point in time of reference image recording and the point in time of localization, which turns out to be highly relevant. For this purpose, we recorded weekly test sequences of similar paths over a period of 24 weeks.

Over the course of this dissertation, we examine state-of-the-art ground image based localization methods on the HD Ground Database.

This chapter is structured as follows: In Section 3.1, we shortly introduce related work, while Section 3.2 introduces the work of Zhang et al. [2019], which presents the only other available ground image databases suited for ground texture based localization: the *Micro-GPS databases*. Afterwards, in Section 3.3, we describe our database, including the employed strategies to data recording

and mapping. Then, Section 3.4 introduces our strategy to generate ground truth query image poses. Finally, we conclude the chapter in Section 3.5 with a general discussion about the application of the introduced databases for the evaluation of ground texture based localization methods.

3.1 Related Work

To the best of our knowledge the *Micro-GPS* databases of Zhang et al. [2019] are the only other publicly available ground image databases suited for ground texture based localization. We introduce them in Section 3.2.

Another publicly available database of ground images was created by Xue et al. [2017]. It contains more than 30000 images of 40 outdoor ground types The database is used to show the effectiveness of differential angular imaging for in-place material recognition. Accordingly, it provides many images of the same places from varying camera angles, but not the area coverage required to examine localization tasks.

Alternatively, Rodriguez and Castano-Cano [2019] propose to generate image data with a virtual camera from simulated vehicle drives over a single high-resolution terrain image, e. g. an image stitching of ground images. This allows to generate a virtually infinite number of different image sequences for training and testing. However, while some image conditions like the camera position, image resolution, motion blur, and lighting can be simulated, this does not allow to examine the effects of actual changes that appear on the ground. In contrast, our HD Ground Database captures changes that appear over time due to wear and tear, as well as due to weather.

3.2 The Micro-GPS Databases

Zhang, Finkelstein, and Rusinkiewicz [2019] present two databases, one that was recorded with a PointGrey CM3 camera (see Figure 3.1), that takes 8-bit gray scale images, and one that was recorded with an iPhone 6 (see Figure 3.2). For both cameras, outdoor and indoor areas are scanned by manually moving a mobile cart on a zig-zag course over the area of interest. The cart is equipped with its own illumination of the ground and it shields the recording area from external illumination. In order to align the recorded images, they are stitched together, based on feature correspondences of overlapping images. Here, the

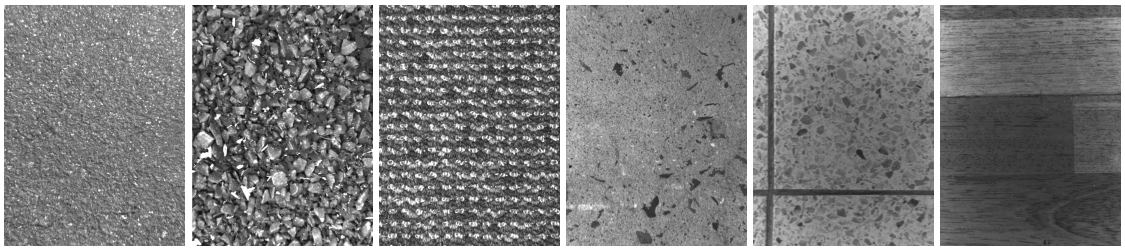


Figure 3.1: Example images of the Micro-GPS database [Zhang et al., 2019] recorded with a PointGrey CM3 camera. From left to right: fine asphalt, coarse asphalt, carpet, concrete, tiles, and wood.

Figure 3.2: Example images of the Micro-GPS database [Zhang et al., 2019] recorded with an iPhone 6. From left to right: granite and tiles

authors proceed in three steps: a) In order to reduce computation time and the effort of having to properly record the entire region at once, the application area is divided into multiple smaller regions of several square meters in size. The images of each region are stitched together by minimizing the squared global reprojection error of feature correspondences; b) connections between the regions are specified manually and they are aligned relatively to each other in another optimization step; c) finally, the individual image poses in the overlap of the smaller regions are optimized in another global optimization step, considering the feature correspondences between images from the region borders, while image poses are kept constant if they are not overlapping with images of other regions.

Subsequently, test images are recorded on (to the best of our knowledge) arbitrary paths on the application areas. For evaluation these images can be used as query images that are to be localized.

Details on the content of the six textures of the Micro-GPS PointGrey CM3 database are presented in Table 3.1, respectively for the two textures recorded with an iPhone 6 in Table 3.2.

In [Schmid et al., 2020a] and [Schmid et al., 2020b] the PointGrey CM3 database was used to compare localization approaches. But for many evaluated localization problems, e. g. localization with prior pose estimate (on all textures but wood), every method reaches close to perfect success rate, raising the question of whether the database covers all challenges of the task. For example, it is

Table 3.1: Details about the content of the Micro-GPS PointGrey CM3 database [Zhang et al., 2017]. Reference images are the images that have been obtained from scans of the application areas. Test images are the images that are to be localized in respect to the reference images. They are provided as separate sequences of consecutively recorded images. Note that the PointGrey CM3 database provides three different application areas with fine asphalt surface, of which we present only the one with the largest number of images, which is also the one that we used for the evaluations in the remainder of this dissertation.

	Fine asphalt	Coarse asphalt	Carpet	Concrete	Tiles	Wood
Area covered m^2	19.76	21.20	17.70	32.68	12.75	41.76
#Reference images	2215	2061	2014	3316	4043	3826
#Test images	4887	3570	8817	7012	1077	1165
#Test sequences	11	9	27	20	3	4

Table 3.2: Details about the content of the Micro-GPS iPhone 6 database [Zhang et al., 2017].

	Granite	Tiles
Area covered m^2	27.52	12.75
#Reference images	1229	1296
#Test images	862	1621
#Test sequences	3	3

not systematically covering ground changes that occur over time. For this and further evaluations, we are introducing the HD Ground Database.

3.3 The HD Ground Database

We present a large database of ground images for the development and evaluation of ground texture based localization methods: the HD Ground Database. It contains reference and test images of eleven textures. Reference images cover the application areas, and were aligned in an image stitching process, similar to that of the Micro-GPS databases of Zhang et al. [2019].

The four main textures (see Figure 3.3) are footpath asphalt, parking place cobblestone, office felt carpet, and kitchen laminate. For these textures, test image sequences were captured systematically by recording a similar trajectory on a weekly basis (detailed timings are given in Section 7.3.3). Additionally, as presented in Figure 3.4, separate sets of trajectories were recorded in quick succession. These trajectories are following quite precisely the same path on the coverage area, which allows to evaluate a teach-and-repeat scenario in which

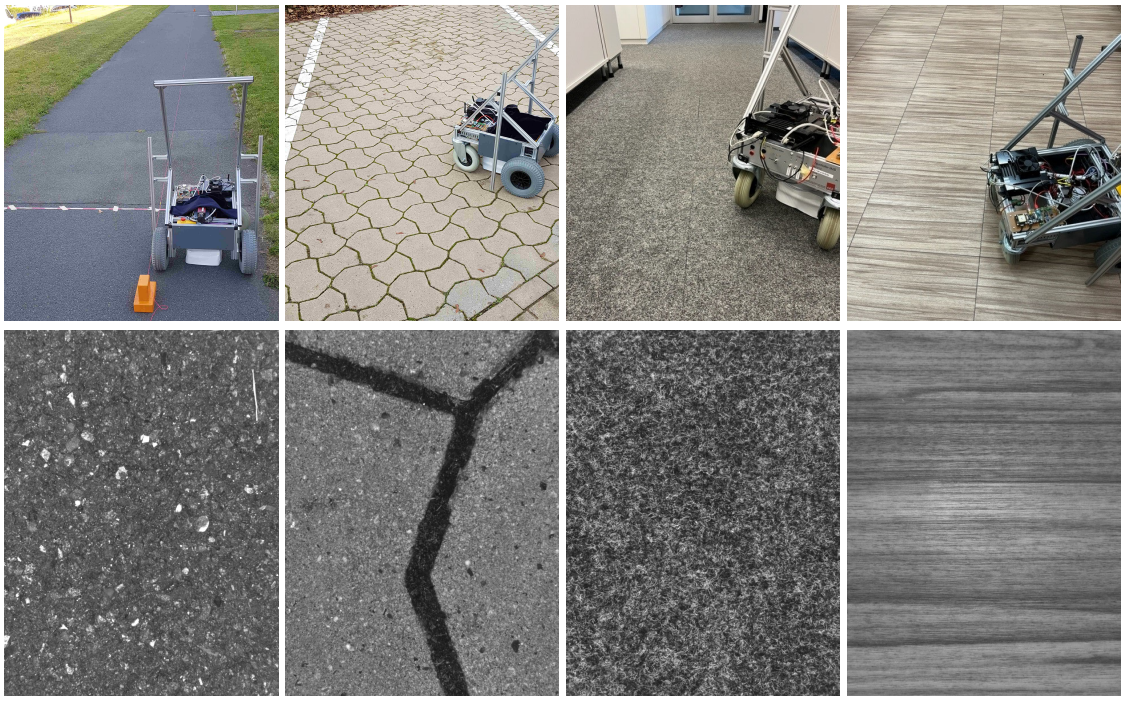


Figure 3.3: Representation of the application area, as well as example images of the four main textures of the HD Ground Database. From left to right: asphalt, cobblestone, carpet, and laminate. This figure is adapted from [Schmid et al., 2022].

Table 3.3: Details of the 4 main textures of the HD Ground Database. Regular sequences are the test sequences that we recorded on a weekly basis.

	Asphalt	Cobblestone	Carpet	Laminate
Area covered m^2	106.12	59.28	90.15	16.18
#Reference images	32251	25337	33456	5812
#Test images	17483	14442	16579	9052
#Regular sequences	12 dry, 9 wet	12 as it is, 12 cleaned	22	22

a robot is steered along a specific path once, and subsequently is supposed to follow the taught path autonomously in both directions. Table 3.3 presents further details for the main textures. Typically, localization methods are adapted to a database or a specific texture through training or parametrization. For this purpose, as illustrated in Figure 3.5, we provide additional *training areas*: a separately recorded square meter for each of the main textures, and additionally a $2\,m^2$ door mat. For six further textures (see Figure 3.6) reference and test images are captured on the same day: terrace pavement ($24.8\,m^2$), garage concrete ($18.2\,m^2$), workroom linoleum ($17.1\,m^2$), bathroom tiles ($3.8\,m^2$), checker plate steel ($3.3\,m^2$), and ramp rubber ($2.8\,m^2$). We call these *generalization textures*, as we will use them to evaluate generalization capabilities.

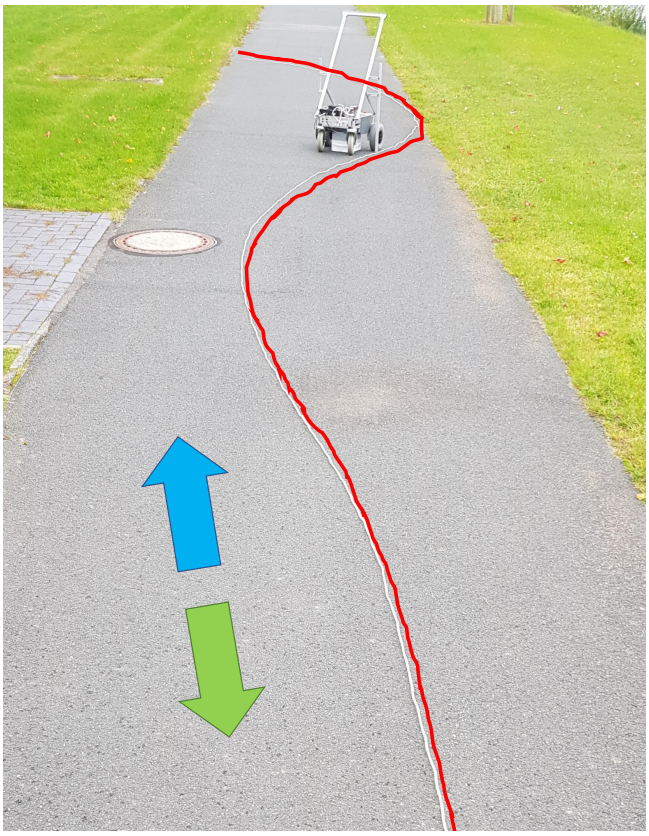


Figure 3.4: Data recording setup on the footpath asphalt application area for the teach-and-repeat scenario. First, a rope (in the image traced by the red line) is put on the ground in an arbitrarily chosen configuration. Then, we move the recording platform along the rope while recording image sequences. The aluminium profiles attached to the sides of the recording platform act as visual guidance for tracking the rope. For two sequences (green arrow, as seen in the image), we follow the rope aligning the left profile with the rope while moving in the direction of the viewer, and for two further sequences (blue arrow), we follow the rope while moving away from the viewer aligning the right profile with the rope.

3.3.1 Setup of the Recording Platform

Our recording platform is a modified RT3-2 VolksBot [Surmann et al., 2008] (see Figure 3.7). The only sensor being used for our database is the ground-facing camera. The recording area is shielded from external lighting and illuminated by a 24 V, 72 Watt LED ring. Pulsed LED lighting is synchronized with the camera exposure, reducing heat generation significantly, allowing us to provide bright illumination during recording, enabling exposure times of only about 0.09 ms, preventing any significant motion blur at our pre-defined operational speeds of up to 5.56 m/s. Images are recorded at a frame rate of 50 Hz. In practice, however, we retain only every fourth frame, because we move the platform

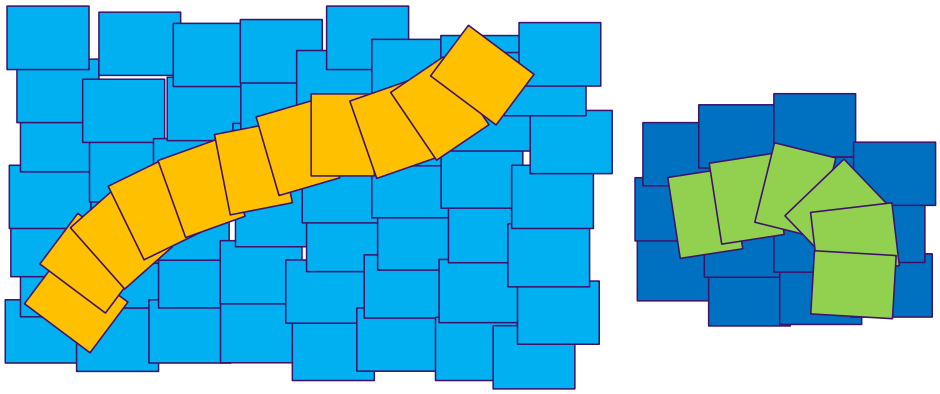


Figure 3.5: Visualization of the types of data provided by the HD Ground Database. Rectangles represent images. Each application area is covered by a set of reference images (light blue). They form the map in which separately acquired query images (orange) are to be located. For the main textures, additional training areas (dark blue) and query images (green) are recorded. This figure is taken from [Schmid et al., 2022].

at most with a quarter of the intended maximum speed. We observe that our lighting creates strong specular reflections on some types of ground texture. Therefore, inspired by Kelly et al. [2007], we add a pair of polarization filters to enable *cross-polarization*. Here, the polarizer in front of the LED ring lets only light waves with a certain polarization pass. The analyzer in front of the camera is orthogonal to the polarizer, and therefore removes exactly the light waves with the polarization that was previously passed. Since specular reflected light keeps its polarization, it is not reaching our camera. This method, as demonstrated in Figure 3.8, effectively prevents specular reflections in the recordings.

It can also help with wet surfaces, as shown in Figure 3.9. However, with this solution a significant part of the emitted light is removed: both polarizations of the light remove about 50% of the respective incoming light. For stable imaging, this should be compensated, but this would require an increase in power consumption for brighter lighting or an increase in exposure time.

In the end, cross-polarization was not employed for the recordings of the HD Ground Database, because the examined textures are not critically reflective.

Some of the most important design parameters for a recording setup with a downward-facing camera are the exposure time τ, the longitudinal length l_{long}, i. e. the image size along the main direction of driving, of the recorded image, and the camera height h. We can derive guidelines for selecting the values of those quantities by considering our requirements for the platform: a vehicle speed of $v_{\max} \leq 20\,\text{km/h} \approx 5.56\,\text{m/s}$ should be supported; for visual odometry, consecutive images should have a longitudinal relative overlap of

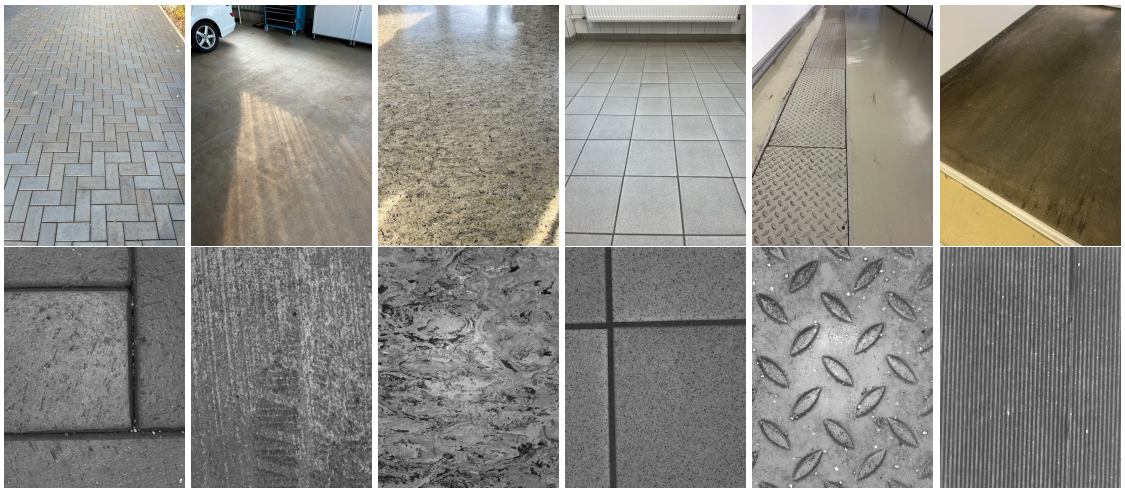


Figure 3.6: Representation of the application area, as well as example images of additional textures of the HD Ground Database. From left to right: pavement, concrete, linoleum, tiles, steel, and rubber. This figure is adapted from [Schmid et al., 2022].

$o_{\text{min}} \geq \frac{1}{3}$; and motion blur, i. e. the traveled distance during exposure b, should be $b_{\text{max}} \leq 0.5\,\text{mm}$ to allow to distinguish between small texture details. Three more constraints are given by the maximum recording speed of our AVT Manta G-235C camera (Sony IMX174 global shutter CMOS sensor) with $f = 50\,\text{Hz}$; the recording opening angle of our lens (Schneider Kreuznach Cinegon 1.4/12-0906) of $\alpha \approx 48\,^\circ$ image diagonal; and the image aspect ratio of $4:3$.

The exposure time is derived from the maximum allowed motion blur b_{max} and the supported vehicle speed v_{max} as

$$\tau = \frac{b_{\text{max}}}{v_{\text{max}}} \approx \frac{0.0005\,\text{m}}{5.56\,\text{m/s}} \approx 0.09\,\text{ms}. \tag{3.1}$$

The longitudinal image length is defined by the vehicle speed v_{max}, the recording frequency f, and the image overlap o_{min}:

$$l_{\text{long}} = \frac{v_{\text{max}} \cdot 1/f}{1 - o_{\text{min}}} \approx \frac{5.56\,\text{m/s} \cdot 0.02\,\text{s}}{1 - 1/3} \approx 0.167\,\text{m}. \tag{3.2}$$

Finally, the camera has to be mounted high enough to capture the diagonal of our coverage area with length l_{long} and width $l_{\text{lat}} = l_{\text{long}} \cdot 3/4$, given the camera opening angle α:

$$h = \frac{0.5 \cdot \sqrt{l_{\text{long}}^2 + (3/4 \cdot l_{\text{long}})^2}}{\tan(\alpha/2)} \approx 0.234\,\text{m}. \tag{3.3}$$

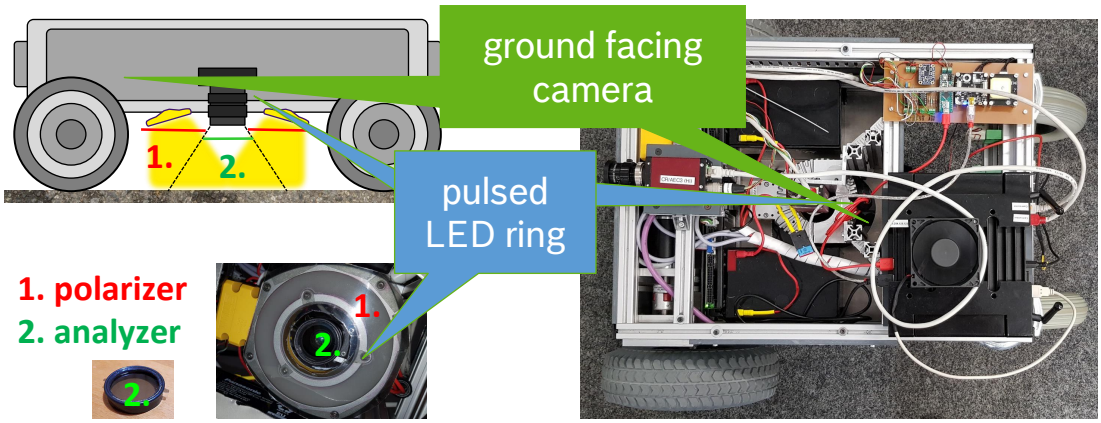


Figure 3.7: An illustration of our recording platform. Image recording is synchronized with a pulsed LED ring for illumination. A combination of polarizer and analyzer in front of the LED ring, respectively the camera, can be used to remove specular reflections. This figure is adapted from [Schmid et al., 2022, accompanying video].

3.3.2 Data Recording

We differentiate three systematic setups of data recording.

- **Initial scanning of the whole coverage area (reference images).** The application area is recorded lane-by-lane, with each lane having a lateral offset of 38 to 50 mm to the previous one. That way images have approximately 2/3 overlap with neighboring images from the previous and next lane, and, because we drive slowly during mapping, also with the previous and next image of the same lane. Accordingly, every point on the ground is covered by about 9 reference images, which allows us to properly align the images during mapping.
- **Recording of regular test sequences**. For each main texture, we define a regular test path. Weekly test sequences are recorded by roughly following the respective paths. For cobblestone, two regular test paths are recorded: one where the area is cleaned before recording and one where it is not. For cobblestone only sequences with mostly dry surfaces are recorded. For asphalt, on the other hand, additional sequences are recorded with weather-caused wet surface.
- **Recording of teach-and-repeat sequences**. A 20 m rope is put in a curved shape on the application area. Then, we closely follow this rope two times in forward and backward driving direction. Five different rope configurations are recorded per texture, one example for the asphalt application area is presented in Figure 3.4.

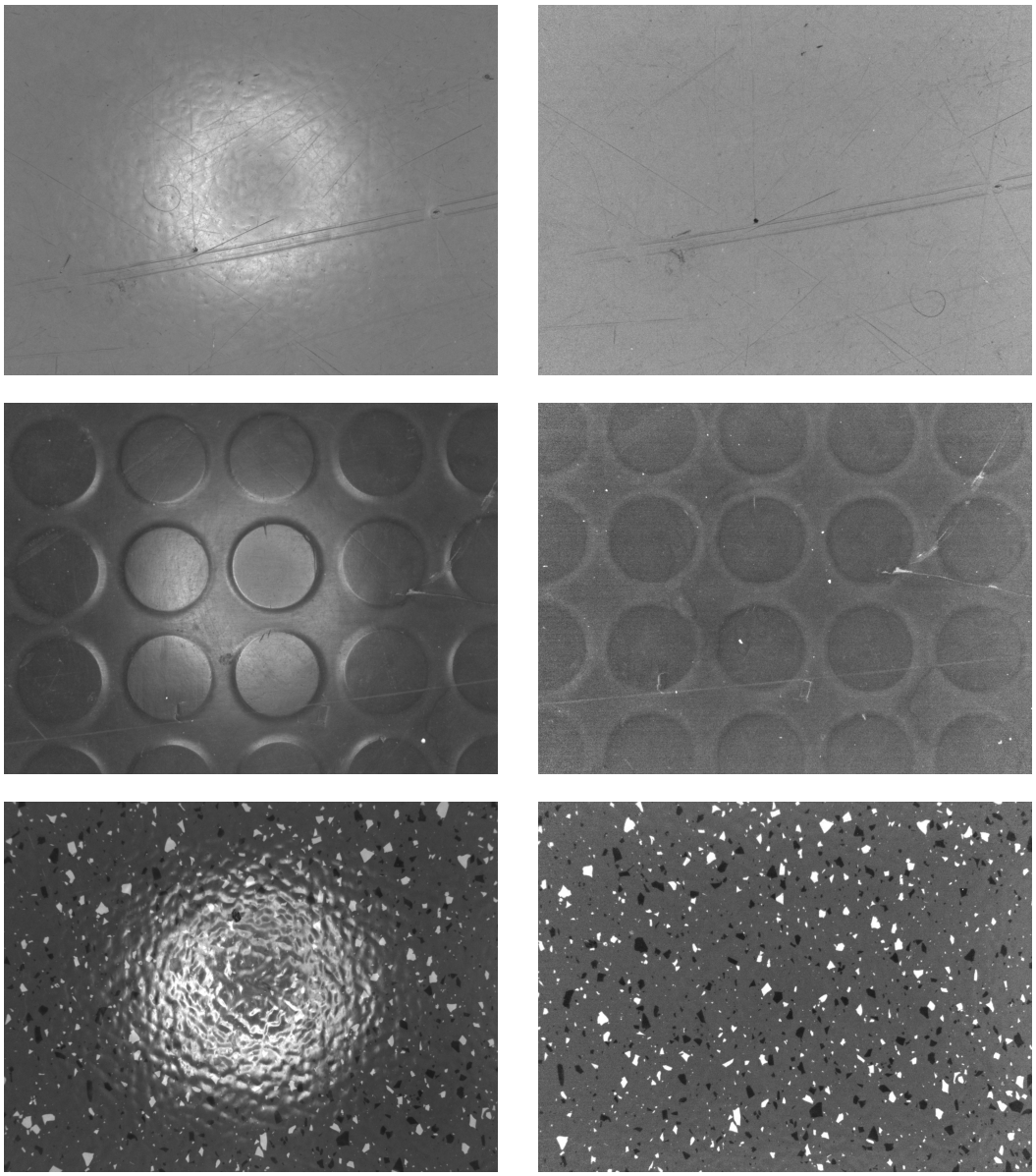


Figure 3.8: Comparison of ground images that have been recorded without (left) and with (right) the cross-polarization filter solution.

For the training areas and generalization textures, test images are recorded on arbitrary paths directly after the initial scanning. We calibrate the camera using a pinhole model with two radial distortion parameters and use the rectified images. Also, we compensate for vignetting (see Figure 3.10), because in our recordings, due to the use of a ring-shaped lighting and a camera with maximum aperture, brightness quickly declines towards the borders. To do this, we compute the average brightness image from 100 recordings of white paper and normalize each image with it.

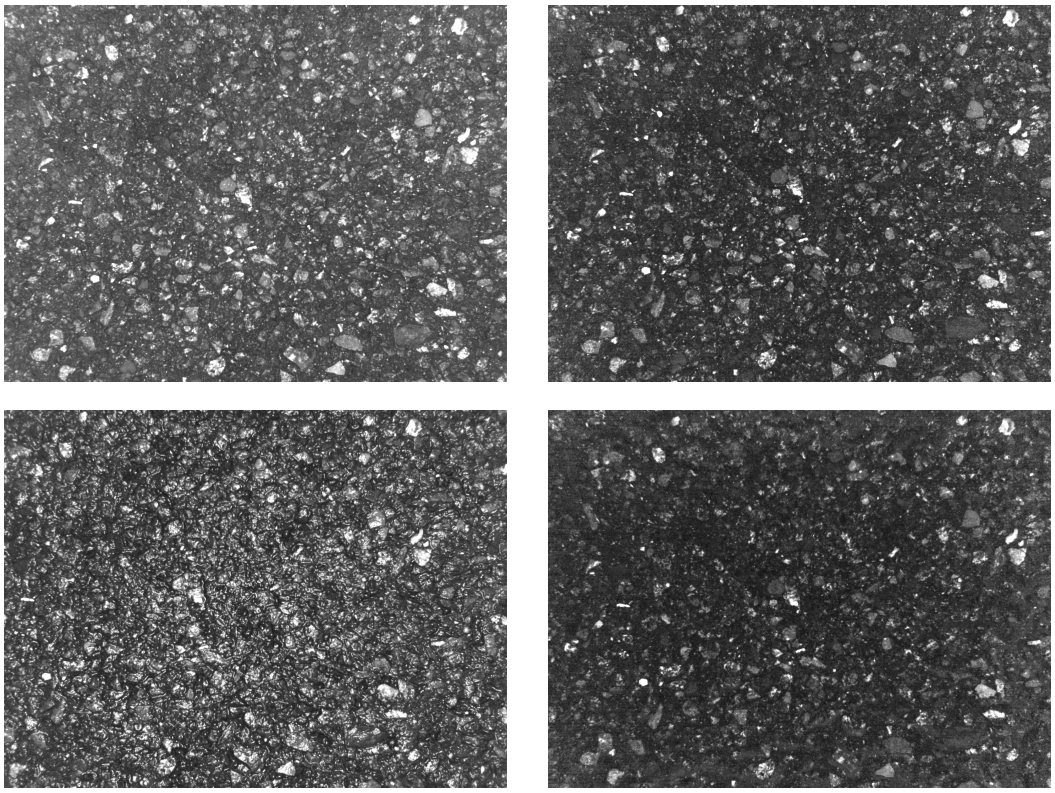


Figure 3.9: Comparison of ground images that have been recorded without (left) and with (right) the cross-polarization filter solution on asphalt. The ground was dry when the images of the top row were recorded, while it was wet during the recording of the images of the bottom row.

3.3.3 Mapping

We create a map for each application and training area. They are created offline with an image stitching process similar to that of Zhang et al. [2019], aligning the reference images in a common map coordinate system.

A first image is put to the origin of the coordinate system and then we compute relative poses of consecutively recorded images. This yields us the initial reference image pose estimates. Relative poses are estimated with a simple feature-based approach, using SIFT features, a ratio test based brute-force nearest neighbor matching strategy, and final RANSAC-based pose estimation. To map the application areas, SIFT parameters are optimized on the respective training areas, while they were initially optimized on the Micro-GPS [Zhang et al., 2019] database to map the training areas. This incremental pose estimation quickly accumulates drift, so only a small set of 5 to 50 images is added to the map at each iteration of the mapping process. Afterwards, we estimate the poses of each image relative to all its (potentially) neighboring images. It is crucial to avoid incorrect estimates at this stage. Therefore, we require that each relative transformation of image n to one of its neighbors (image m) is

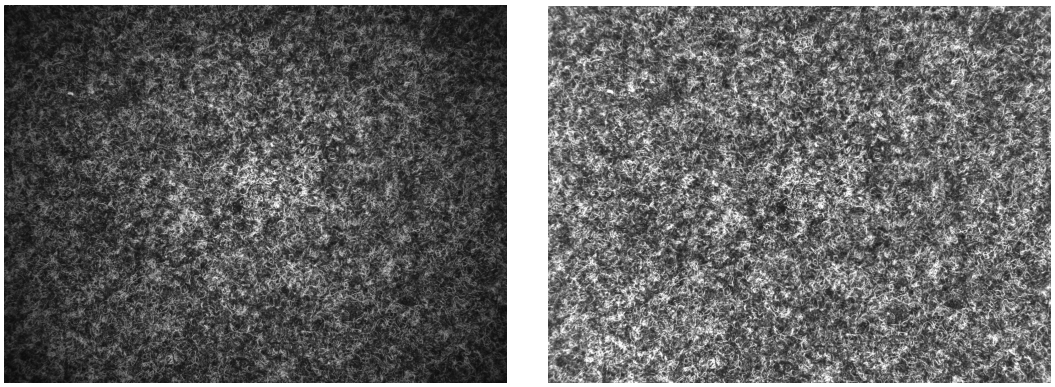


Figure 3.10: Comparison of carpet ground images before (left) and after (right) rectification and compensation for vignetting.

confirmed by the relative transformation of the $(n-1)$-th or the $(n+1)$-th image to image m. Let $[\mathbf{R}, \boldsymbol{t}]_n^m$ denote the transformation from image n to image m, consisting of a rotation $\mathbf{R}$ and a translation $\boldsymbol{t}$, then

$$[\mathbf{R}, \boldsymbol{t}]_n^m \approx [\mathbf{R}, \boldsymbol{t}]_{n-1}^x [\mathbf{R}, \boldsymbol{t}]_n^{n-1}, \tag{3.4}$$

is required, or alternatively

$$[\mathbf{R}, \boldsymbol{t}]_n^m \approx [\mathbf{R}, \boldsymbol{t}]_{n+1}^x [\mathbf{R}, \boldsymbol{t}]_n^{n+1}. \tag{3.5}$$

Furthermore, we require the number of RANSAC inliers to exceed 100, which is an empirical threshold that depends on the employed feature extractor and its parametrization. Unconfirmed image pose relations are discarded.

At the final step of each mapping iteration, the set of all reference image poses $\{[\mathbf{R}, \boldsymbol{t}]\}$ is jointly optimized, considering pairs of corresponding features (f_i^k, f_j^k) between all pairs of neighboring images (i, j), for which the relative pose estimate was confirmed. Similar to Zhang et al. [2019], we formulate the optimization as a non-linear least-squares optimization problem in Ceres[1] [Agarwal et al., 2016], using the loss function:

$$\mathrm{E} = \min_{\{[\mathbf{R},\boldsymbol{t}]\}} \sum_{(i,j)} \sum_{(f_i^k, f_j^k)} ([\mathbf{R}, \boldsymbol{t}]_i^{\mathrm{M}} \cdot f_i^k - [\mathbf{R}, \boldsymbol{t}]_j^{\mathrm{M}} \cdot f_j^k)^2. \tag{3.6}$$

With $[\mathbf{R}, \boldsymbol{t}]_i^{\mathrm{M}}$ denoting the transformation mapping image i into the map M.

In contrast to Zhang et al. [2019], we consider only correspondences that are part of the consensus set of the RANSAC-based pose estimation. This means that we can expect all considered feature correspondences to be correct, and

[1] http://ceres-solver.org/

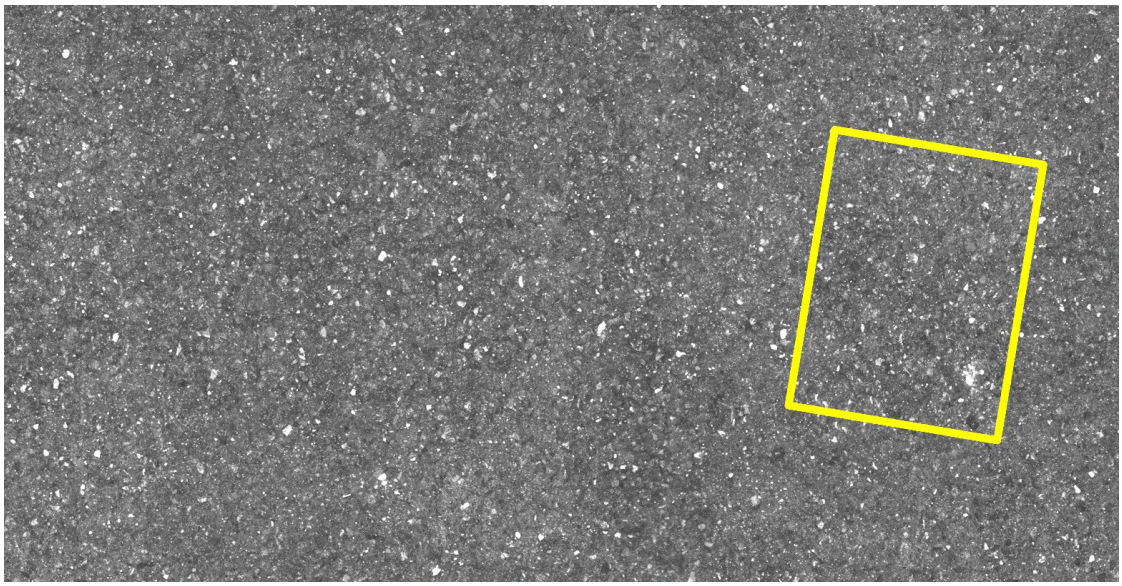


Figure 3.11: A small section from an image stitching of ground images with asphalt texture. The yellow rectangle corresponds to a single image. This figure is adapted from [Schmid et al., 2022].

that a set of two correspondences (each representing a pair of corresponding points between query image and map) is sufficient for a near to full description of the given constraints of an image pose relation, reducing the size of the optimization problem.

In order to create an image stitching of the mapped reference images for visualization, pixel gray values of regions contained in multiple reference images are computed as an average value of the corresponding pixels of all overlapping images. The correctness of our maps is then confirmed by visually inspecting the stitched maps. We observe only small amounts of smearing artifacts. Otherwise, image transitions are smooth as in Figure 3.11 and Figure 3.12.

3.3.4 Comparison with existing databases

One of the most important novel aspects of our database is the recording of regular test sequences for a systematic evaluation of localization performance over time. Also, we enable the evaluation of a teach-and-repeat scenario and our database is larger than existing ones. Table 3.4 compares the sizes of HD Ground with the Micro-GPS databases. The largest coverage area recorded for the HD Ground Database is 2.5 times larger than that of the Micro-GPS databases (41.76m^2 of wood for Micro-GPS compared to 106.18m^2 of asphalt for HD Ground). Larger areas can be used to evaluate the effect of visual aliasing when considering a larger number of reference images during localization.

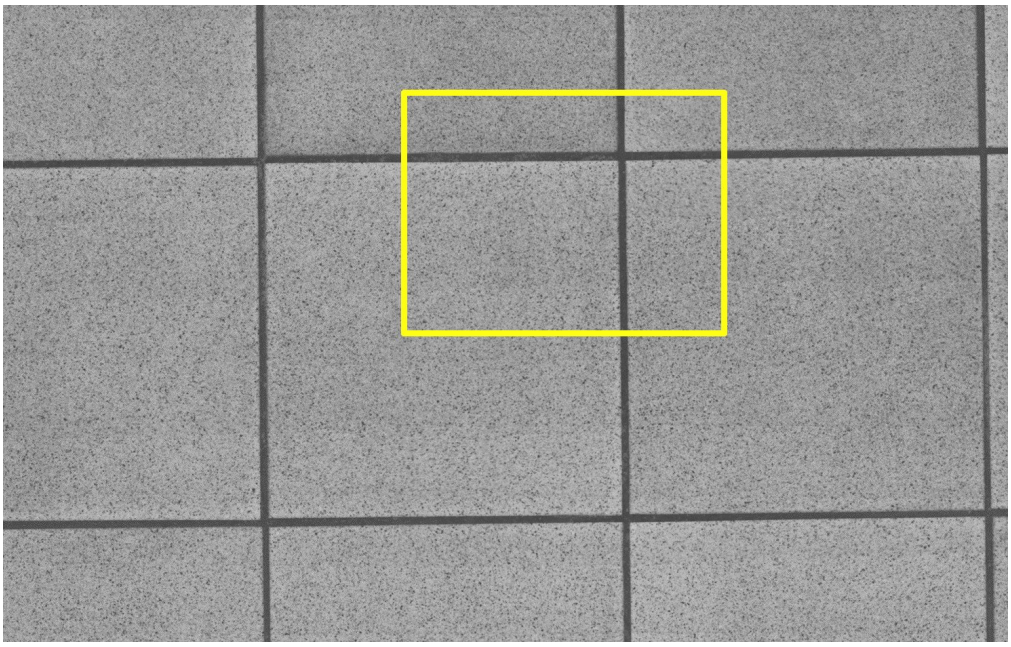


Figure 3.12: A small section from an image stitching of ground images with tiles texture. The yellow rectangle corresponds to the area covered by a single image. This figure is adapted from [Schmid et al., 2022, accompanying video].

Table 3.4: A comparison of the M(icro)-GPS database of Zhang et al. [2019] with our HD Ground Database.

Database	Total area covered	Largest single area	#Reference images	#Test images	#Textures	Resolution	mm / pixel
M-GPS PointGrey	145.85 m^2	41.76 m^2	23 487	28 929	6	1288×964	0.16
M-GPS iPhone 6	40.27 m^2	27.52 m^2	2 525	2 483	2	1280×720	NA
HD Ground	347.73 m^2	106.12 m^2	129 965	71 463	11	1600×1200	0.1

In this context, visual aliasing means that different places have similar visual appearances, leading to confusion during localization.

While Micro-GPS provides a minimal set of reference images covering the application area, we provide overlapping reference images. This means, for example, that our asphalt dataset, with 32251 images, contains more than 8 times as many reference images as the wood dataset of Micro-GPS, with 3826 images, while covering only a 2.5 times larger area. For instance, having overlapping images available, a localization method could reduce its memory footprint storing only those features that consistently appear in the overlap of multiple reference images, as suggested by Schmid et al. [2020b].

Our images present the ground at a higher resolution which allows to examine the extent to which this is beneficial. Also, our exposure time of about 0.09 ms

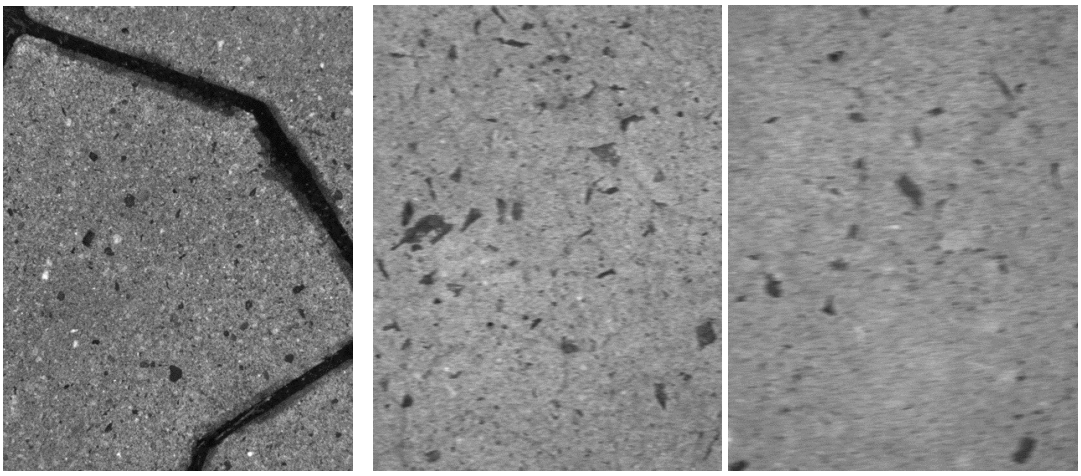


Figure 3.13: As it can be seen in the left image from the HD Ground Database, recording images with an exposure time of about 0.09 ms prevents motion blur from being a problem. The other two images are taken from the Micro-GPS database [Zhang et al., 2019]. According to the authors, they have been recorded with an exposure time of between 3 and 5 ms. While both images show significant motion blur, it depends on the speed of the vehicle during recording. The center image was recording while driving slowly and the right one while driving faster. This figure is adapted from [Schmid et al., 2022, accompanying video].

reduces motion blur compared to the Micro-GPS database with exposure times of 3–5 ms [Zhang et al., 2017] as can be seen in Figure 3.13.

3.4 Query Image Ground Truth Poses

For reference images, we are using the resulting poses in the map coordinate system from the image stitching process as ground truth. This is not possible for query images of the independently recorded test sequences. However, for some evaluations it is required to have ground truth poses of query images, e. g. for parameter optimization and localization with prior.

This is why we employ a procedure similar to that proposed by Zhang et al. [2019] to generate query image ground truth poses for our evaluations on the Micro-GPS databases as well as for our HD Ground Database. Here, an estimated pose is considered to be correct if it is confirmed by a second pose estimate. The first pose estimate is computed using a method for global map-based localization, like Micro-GPS [Zhang et al., 2019], StreetMap [Chen et al., 2018], or one of our own methods that are developed in this dissertation. Alternatively, we employ a method for local map-based localization that takes

advantage of an available query image pose prior to generate this initial query image pose estimate. This is possible if, for example, we already generated a ground truth pose for the previous image of the test sequence. A second pose estimate of the query image, that we use to confirm or decline the first one, is determined with a relative localization method. Here, we follow the suggestion of Zhang et al. [2019] of using a simple SIFT-based approach, which is the same approach we use in the mapping phase of the HD Ground Database, using brute-force nearest neighbor feature matching, a ratio test for outlier rejection, and RANSAC-based pose estimation, to estimate the query image pose in respect to the closest reference image according to the first pose estimate. The original pose estimate is confirmed if it is close to the second one (less than 4.8 mm in distance and 1.5 ° in orientation, as defined by Zhang et al. [2019]).

We repeat this procedure, using several localization approaches to generate the first pose estimates, until a global pose estimate is successfully confirmed. Then, we store the confirming pose estimate (the one computed with the simple SIFT-based approach) as ground truth pose for the image.

3.5 Discussion

This chapter introduced the Micro-GPS databases and our own database: the HD Ground Database. Both aim to solve the Problem 1: the initial scanning of the application area.

The Micro-GPS databases are sufficient for the evaluation of typical localization tasks like map-based localization with and without prior, as well as relative localization, on some of the most common ground types. But, only the HD Ground Database enables a systematic examination of more challenging scenarios, e. g. with significant time intervals between the initial scanning of an application area and subsequent localization, or with significantly larger numbers of reference images, or with a greater variety of ground types to examine generalization capabilities.

A caveat of all presented databases is the missing availability of actual ground truth poses. Reference image poses are determined in the image stitching process, which can take only local consistency of the retrieved feature correspondences into account, while small errors can still accumulate to a significant global drift between image poses in the map and their actual poses in the world. Similarly, we can confirm only the local consistency of query image pose estimates. A solution to this problem would be to measure image poses during

recording with an external reference system. However, to evaluate millimeter-accurate positioning, as it is possible with ground texture based localization, we demand highly accurate ground truth poses, which would require high-precision measuring equipment available at all recorded application areas, significantly increasing the effort and cost of data recording. Anyway, due to the local consistency of the image poses, the available ground truth is sufficient to examine the performance of map-based localization approaches.

The Micro-GPS database recorded with PointGrey CM3 camera will be the main database for evaluation of localization performance in the remainder of this dissertation, because the HD Ground Database was the last of my projects of my doctoral period to be completed. Nevertheless, for several projects, we performed additional evaluations on the HD Ground Database and present them in the corresponding chapters.

4 Local Visual Features for Ground Texture Based Localization

Contents of this chapter were partially published in [Schmid, Simon, and Mester, 2019].

State-of-the-art methods to ground texture based localization employ feature-based localization [Swank, 2012, Nakashima et al., 2019, Chen et al., 2018, Kozak and Alban, 2016] that relies on the extraction of similar features from varying views of the same location. While several feature extraction methods were evaluated in these works, this survey is an extension. We evaluate additional methods for feature detection (e. g. AKAZE [Alcantarilla et al., 2013] and LIFT [Yi et al., 2016]) and feature description (e. g. DAISY [Tola et al., 2010] and LATCH [Levi and Hassner, 2016]), and we consider different techniques for keypoint selection (Non-Maximum Suppression (NMS), Adaptive Non-Maximum Suppression (ANMS), and bucketing).

This work contributes an extensive survey using an elaborate evaluation framework for ground texture based localization performance. For this purpose, we investigate the task of finding corresponding image regions in pairs of overlapping ground images. We examine relevant synthetic transformations of ground images, perform pose estimation in respect to separately taken ground images, and introduce appropriate performance indicators to evaluate keypoint detector performance on ground images.

In this chapter, we are concerned with map-based localization (Problem 3), but instead of considering large maps with thousands of reference images, the set of references images $\mathcal{R}$ consists for each evaluated localization attempt of a single overlapping image with known relative transformation to the query image. Correspondingly, the employed map object is constructed simply by extracting features of that single reference image.

Section 4.1 summarizes other surveys of features for ground images. Then, Section 4.2 introduces the approaches evaluated in this survey. Sections 4.3 and 4.4 describe and evaluate our experiments. Finally, Section 4.5 concludes

the chapter with a summary of the presented content, and a discussion of the gained knowledge.

4.1 Related Work

We present existing surveys of features for ground texture based localization.

Zhang et al. [2019] evaluate the use of SIFT [Lowe, 2004], SURF [Bay et al., 2006], ORB [Rublee et al., 2011], and HardNet [Mishchuk et al., 2017] for their ground texture based localization pipeline called Micro-GPS. They reduce the descriptor dimensionality using Principal Component Analysis (PCA), match descriptors with an ANN search structure, employ the voting procedure (Section 2.3) for outlier rejection, and finally estimate the query image pose in a RANSAC procedure. The authors receive the best results for keypoint objects and descriptors computed with SIFT. In a follow-up paper [Zhang and Rusinkiewicz, 2018], the authors develop a fully convolutional neural network trained on ground texture images that achieves higher repeatability than SIFT, but has increased computational cost.

Kozak and Alban [2016] evaluate combinations of detector and descriptor methods on pairs of partially overlapping ground texture images, measuring the number of correctly matched keypoint objects. Features are matched using nearest neighbor matching with cross-check, i. e. the nearest neighbor property is checked in both directions. They find the combination of CenSurE [Agrawal et al., 2008] keypoint objects and SIFT descriptors to lead to the largest number of successfully matched features. Pairings of CenSurE with ORB descriptors, as well as SIFT detector with SIFT descriptor, also show good performance. FAST [Rosten and Drummond, 2006], SURF, and GFTT [Shi and Tomasi, 1994] keypoint objects, as well as descriptors provided by BRISK [Leutenegger et al., 2011], FREAK [Alahi et al., 2012], or SURF, present significant weaknesses for at least one of the three evaluated road surface texture types: worn asphalt, dark asphalt, and concrete.

Otsu et al. [2013] investigate the suitability of different keypoint detectors for visual odometry from ground texture. They evaluate Harris [Harris and Stephens, 1988], GFTT, and FAST corner detectors as well as the scale-space detectors SIFT, SURF, and CenSurE. The authors identify that none of the detectors is suited for all situations that occur in the employed desert landscape image datasets. Therefore, they propose to switch between detectors dependent on the terrain.

This chapter's survey extends the prior work. We evaluate the computation of keypoint objects separately like in [Otsu et al., 2013, Zhang and Rusinkiewicz, 2018], but also pair them with varying methods for keypoint selection and feature description. In addition to the number of correctly matched keypoint objects, which was considered in [Kozak and Alban, 2016], we evaluate the repeatability of keypoint objects and their spatial distribution, the classification precision of feature matches, measuring the ratio of correct matches to the total number of proposed matches, and the pose estimation success rate. In comparison to [Zhang et al., 2019], we evaluate a larger variety of methods for detection and description. Furthermore, we evaluate performance on synthetically transformed images as well as on separately taken images. In case of the separately taken images, we evaluate sequentially taken image pairs as they occur during incremental *relative localization*, where the transformation is close to a pure translation, and image pairs taken at different times from independent poses as they occur for *absolute localization*, which present potentially strongly divergent orientation.

4.2 Evaluated Approaches to Feature-Based Localization

Previously (Section 2.2), we divided the task of feature-based localization into the 5 subtasks: a) keypoint detection, b) keypoint selection, c) feature description, d) feature matching, e) pose estimation.

For the first three tasks, we examine a range of popular approaches available in OpenCV [Bradski, 2000], as well as LIFT [Yi et al., 2016], a deep learning approach to the extraction of local visual features. For matching and pose estimation, we revert to standard techniques. For matching, we compute the Euclidean distance for real-valued descriptors and the Hamming distance for binary ones. Then, features are matched with nearest neighbor matching and the *ratio test constraint* as suggested by Lowe [2004]. This means that for each query image feature the two reference image features with closest descriptors to that of the query image feature are found. The closer one is suggested as a match if its distance to the query image feature is smaller than that of the second closest one multiplied with a pre-defined ratio test factor. Finally, we estimate the relative poses of query images using the proposed feature matches and RANSAC-based estimation of a 2D Euclidean pose transform.

4.2.1 Evaluated Keypoint Detectors

We evaluate the following keypoint detection approaches, which have been introduced in Section 2.1: Good Features To Track (GFTT) [Shi and Tomasi, 1994], FAST [Rosten and Drummond, 2006], AGAST[Mair et al., 2010], Harris-Laplace (H.L.)[Mikolajczyk and Schmid, 2002], SIFT [Lowe, 2004], SURF [Bay et al., 2006], CenSurE [Agrawal et al., 2008], BRISK [Leutenegger et al., 2011], ORB [Rublee et al., 2011], AKAZE [Alcantarilla et al., 2013], MSER [Matas et al., 2004], MSD [Tombari and Di Stefano, 2014], and LIFT [Yi et al., 2016].

4.2.2 Evaluated Keypoint Selection Methods

We hypothesize that one of the more difficult situations for ground texture based localization is a case in which larger image areas are weakly textured, i. e. areas without or with only a few contrastive image patches. To find potential correspondences between query and reference images in such a case, it is still necessary to extract a sufficiently large number of keypoint objects. Detection parameters should be chosen with respect to this case or need to be adapted texture dependently. Using the same parametrization for all ground types possibly being encountered is desirable, but, a feature detector that is parametrized to retrieve a sufficient number of features from weakly textured images may retrieve a large number of features on other images. Therefore, in order to limit the required processing time for localization, keypoint selection, i. e. the reduction of considered features, has an important role for some feature extraction methods on ground images.

One approach to keypoint selection is NMS, where only the n keypoint objects with largest response value of their corresponding detection criterion are kept. In order to improve the spatial distribution of keypoints, NMS can be combined with *bucketing* [Kitt et al., 2010], where keypoint objects are detected independently for areas defined by a regular grid. An alternative approach is adaptive non-maximum suppression ANMS, where keypoint objects with strong responses suppress keypoint objects in a local neighborhood.

4.2.3 Evaluated Feature Description Methods

We examine description methods generating real-valued descriptors: SIFT [Lowe, 2004], DAISY [Tola et al., 2010], SURF [Bay et al., 2006], and LIFT [Yi et al., 2016]; and we examine description methods generating binary descriptors:

Table 4.1: The experimental setups examined in this survey. For the task of evaluating keypoint detection, we examine synthetically transformed images, and evaluate performance using keypoint repeatability, our novel ambiguity indicator, and the $< n$ KPs indicator, which counts the number of images for which less than n features are extracted. Feature matching is also evaluated with synthetically transformed ground images, based on the number of obtained correct matches, and the classification precision of the proposed correspondences. The final pose estimation task is examined both on synthetically transformed images and actually overlapping, independently recorded ground images. Here, we examine the success rate, i. e. the ratio of localization attempts for which the differences between estimated pose and ground truth pose is small (below 4.8 mm and 1.5 °).

Task	Transformation	Performance indicators
Keypoint detection	Synthetic	Repeatability, Ambiguity, $< n$ KPs
Feature matching	Synthetic	Number of correct matches, Precision
Pose estimation	Synthetic & Real	Success rate

BRIEF [Calonder et al., 2010], ORB [Rublee et al., 2011], BRISK [Leutenegger et al., 2011], FREAK [Alahi et al., 2012], LATCH [Levi and Hassner, 2016], and AKAZE [Alcantarilla et al., 2013]. All of which were introduced in Section 2.1.

4.3 Experimental Setups

Our experimental framework consists of three separate experiments, which are summarized in Table 4.1. The first experiment examines keypoint detection on synthetic transformations, the second one feature matching on synthetically transformed images, and the third experiment examines pose estimation using both synthetic transformations and separately recorded, partially overlapping, ground image pairs.

For synthetic transformations, correct feature matches are known and performance can be evaluated in regard to specific types of image modifications. We evaluate geometric and photometric transformations. Typical photometric transformations that should be considered are Gaussian noise and illumination changes. For this purpose, we add noise that is independent and identically distributed (i.i.d.) and zero-mean. For illumination changes, we employ gamma correction: pixel values g are modified as: $g_{\text{out}} = \text{round}(g_{\text{max}} \cdot (\frac{g_{\text{in}}}{g_{\text{max}}})^{\gamma})$, where $g_{max} = 255$. For both photometric transformations, pixel intensity values are clipped at 0 and 255. Additionally, two geometric transformations are relevant

when using downward-facing cameras: rotation and translation. Rotated images are computed using bicubic interpolation. In case of translation, an image mask determines a section of the image from which features are extracted. This mask is translated by a discrete number of pixels for testing, as illustrated in Figure 4.1. Accordingly, different image sections with specified intersections are evaluated. For this evaluation, only keypoint objects from the intersection between reference mask and query mask are considered.

For separately taken images, it is difficult to obtain sufficiently accurate ground truth in order to determine which feature matches are correct. However, they allow us to examine localization performance with its difficulties that occur during application in the real world. We examine image pairs that are recorded in direct sequence, which represent the challenges of incremental localization, and we examine image pairs that have been recorded at different times and from independent views, which represent the challenges of absolute localization.

4.3.1 Keypoint Detection

We use synthetic transformations to examine whether the same keypoint objects are found in pairs of reference and query images. Pairs of keypoint objects from reference and query image are considered to match and therefore to represent the same location if the IoU of their corresponding image patches in the reference coordinate frame is greater than 0.5.

As performance metric, we evaluate the keypoint **repeatability** introduced by Mikolajczyk et al. [2005]. It measures the proportion of keypoint objects from the query image that were also found in the reference image. Additionally, we introduce two novel performance indicators: ambiguity and $< n$ KPs. With **ambiguity** we address the problem of the repeatability metric that it does not penalize ambiguous keypoint correspondences. This problem occurs if a keypoint object from the query image has multiple valid matches in the reference image, which happens for example if keypoints are densely clustered. We compute ambiguity as the mean number of valid matches of the query image keypoint objects with at least one valid match. Therefore, ambiguity ≥ 1.0. An ambiguity greater 1.0 suggests that the repeatability score is inflated by ambiguous keypoint matches. The second new performance indicator that we introduce is $< \boldsymbol{n}$ **KPs**, which measures how often fewer than n keypoint objects are found in an image. This is of interest, as having only few keypoint objects increases the risk of failure for feature-based localization.

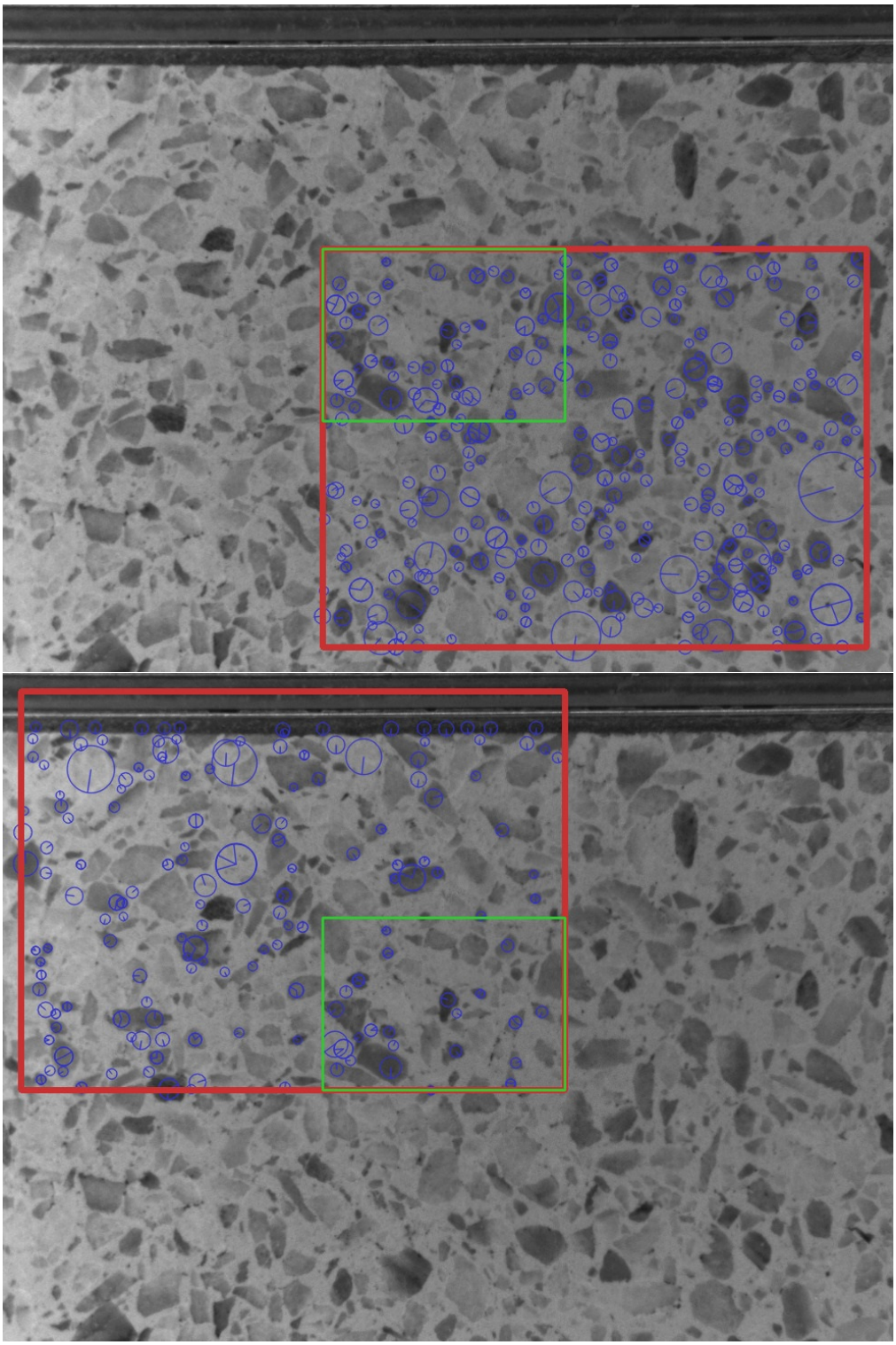


Figure 4.1: SIFT keypoint objects [Lowe, 2004] on a synthetically translated image pair. The keypoint objects are depicted as blue circles, the image sections from which keypoint objects are extracted are bounded by red rectangles, and the intersection of reference image section and translated image section is depicted as green rectangle. In this case, the translation results in an Intersection over Union (IoU) of 0.1925. This figure is adapted from [Schmid et al., 2019, supplementary].

4.3.2 Feature Matching

In order to evaluate whether the obtained features are suited for the localization pipeline, we examine feature matching performance. We evaluate the **number of correctly matched features** and compute the the matching **precision**, based on the number of correct matches (inliers) $|\mathcal{I}|$ and the number of incorrect matches (outliers) $|\mathcal{O}|$: precision $= \frac{|\mathcal{I}|}{|\mathcal{I}|+|\mathcal{O}|}$.

4.3.3 Pose Estimation

Adopting the thresholds of Zhang et al. [2019], we consider pose estimates to be correct if their distance to the *ground truth* is less than $d_t = 4.8\,\text{mm}$ and if their absolute orientation error is less than $o_t = 1.5\,^\circ$. We evaluate pose estimation performance using the **success rate** metric, which is computed as the ratio of the number of correct pose estimates to the number of incorrect ones.

4.4 Evaluation

For evaluation of our experimental framework, we use the six textures of the ground image database of Zhang et al. [2019] (see 3.1), recorded with a gray-scale PointGrey CM3 camera. An E3-1270 Intel Xeon CPU at 3.8 GHz is employed for compuation. We randomly select 3 images per texture to be used exclusively for parameter optimization, 100 for the evaluation with synthetic transformations, and 100 actual reference-query image pairs each for incremental and absolute localization tasks. We observed no significant performance variations using more query images. Our strategies for parameter optimization, and the obtained parameter settings can be found in the appendix Section A.1.

We make use of OpenCV 4.0 [Bradski, 2000] implementations for most of the evaluated methods for feature detection and description. Due to bad performance of the ORB implementation of OpenCV, we use its implementation that comes with ORB-SLAM2 [Mur-Artal and Tardós, 2017]. The implementation and the trained network weights of LIFT are provided by the authors, who claim to achieve good generalization performance even without domain specific training samples [Yi et al., 2016]. We exclude ORB and LIFT from the evaluation on synthetic transformations as their implementations do not allow to restrict the search space using a detection mask, which is what we require for the evaluation of synthetic translation. For feature matching, we find most similar reference descriptors and filter them with the ratio test constraint with a factor

of 0.7. Poses are estimated using RANSAC-based estimation of a Euclidean transformation with a maximum of 2000 iterations and the error threshold applied in [Zhang et al., 2019] of 3.0 pixels.

Synthetic transformations are parametrized as follows: for rotation, angles between 0 ° and 180 °; for translation, we evaluate IoUs of reference mask and query mask between 0.2 and 1.0. Gaussian i.i.d. noise is zero-mean with standard deviation between 0.0 and 40.0; illumination values are changed non-linearly using a gamma between 0.1 and 3.0. When presenting results from synthetic transformations, performance indicators are averaged with equal contribution of the results of each transformation type.

4.4.1 Evaluation of Selector-Detector Pairings

We examine the repeatability of keypoint detectors using the keypoint selection methods introduced in Section 4.2.2 to reduce the number of keypoint objects to 1000. Respectively, if the keypoint detection method allows to specify the desired number of keypoint objects, we set this parameter to 1000. For keypoint selection with ANMS, we use Suppression via Square Covering [Bailo et al., 2018] with a tolerance of 20%. For bucketing, we received good results for non-square buckets, using a grid of 8 rows and 6 columns. For each grid cell, 21 keypoint objects are selected using NMS resulting in a maximum of 1008 keypoint objects per image. The evaluation of a single image without the employment of keypoint selection takes us several days, due to the large number of retrieved keypoint objects of some detection approaches (especially AGAST with an average of more than 200 000 keypoint objects per image) and due to the large number of applied synthetic transformations in the experiment conducted. Which is why for this particular experiment, we evaluate a single query image per type of ground texture, and apply the synthetic transformations to it. In addition to the repeatability scores, Table 4.2 presents the average number of keypoint objects before selection. MSER does not provide a keypoint response measure, and is therefore not well suited to be used with a selection method. Together with FAST, AGAST, and BRISK, MSER has significantly better repeatability without selection. In order to select MSER keypoints without a keypoint response measure anyway, we use the order of extracted keypoint objects as substitution for the response measure. This means that the first found maximally stable extremal regions, which are the ones with lowest intensity values, are considered to have the largest response. With this workaround MSER still achieves a surprisingly large repeatability of 51% using NMS and 73% using

Table 4.2: Evaluation of varying keypoint detection methods for the number of keypoint objects (#KPs) before selection (w/o selection), and their achieved repeatability if paired with one of the evaluated keypoint selection strategies: Non-Maximum Suppression (NMS), Adaptive Non-Maximum Suppression (ANMS), and bucketing [Kitt et al., 2010].

Detector	#KPs before selection	Repeatability w/o selection	NMS	ANMS	Bucketing
AKAZE	10199	0.81	**0.82**	0.41	0.74
SIFT	755	0.82	**0.82**	**0.82**	0.77
SURF	7271	0.80	**0.82**	0.65	0.74
CenSurE	6434	0.83	**0.76**	0.39	0.70
MSD	6484	0.59	**0.76**	0.51	0.68
H.L.	839	0.76	**0.76**	**0.76**	0.68
MSER	15238	0.94	0.51	**0.73**	0.52
BRISK	58050	0.84	**0.71**	0.25	0.64
GFTT	894	0.69	**0.69**	**0.69**	0.29
AGAST	225361	0.93	**0.64**	0.22	0.57
FAST	52112	0.78	**0.64**	0.26	0.56

ANMS. We find MSER to be the only detector that performs best with ANMS. For all other detectors, we use NMS in the following. The repeatability of AKAZE, SURF and especially MSD is increased when using keypoint objects selected by NMS instead of all available keypoint objects. This means that keypoint objects that have been assigned low values of the keypoint response measures of these detectors are indeed non-repeatable and are rightly removed by NMS.

In a next step, we evaluate the best performing detector-selector pairings using the full 100 reference images for testing. Averaged results are presented in Table 4.3. For the $< n$ KPs performance indicator, measuring the ratio of images for which less than n keypoint objects are detected, we set n to 100, as we noticed that localization success is low with fewer keypoint objects. For most detectors, we were able to find parameters that allow to retrieve at least 100 keypoint objects from almost all images. But, AKAZE, H.L., and GFTT still find fewer keypoint objects on at least 4% of the images. This problem occurs almost exclusively on wood texture images. H.L. extracts less than 100 keypoint objects on 49% of wood images, GFTT on 28% and AKAZE on 23%. SIFT, AKAZE, and SURF have with 83% to 84% the best repeatability. However, SIFT has a large ambiguity score of 1.50. This is due to SIFT generating multiple keypoint objects with different orientations at the same position in case that there are multiple strong orientations in the histogram of gradient directions, which creates clusters of keypoint objects with ambiguous correspondences in the transformed image.

Table 4.3: Keypoint detectors are paired with their best performing keypoint selection method, considering Non-Maximum Suppression (NMS), Adaptive Non-Maximum Suppression (ANMS), and bucketing [Kitt et al., 2010]. These pairings are evaluated for the < 100 KPs indicator, which is the number of images for which less than 100 features have been extracted, their repeatability, ambiguity, and computation time. The evaluation is done with synthetically transformed images of the Micro-GPS database [Zhang et al., 2019].

Selector	Detector	< 100 KPs	Repeatability	Ambiguity	Comp. time (s)
NMS	AKAZE	0.05	0.84	1.00	0.40
NMS	SIFT	0.00	0.83	1.50	3.27
NMS	SURF	0.01	0.83	1.06	0.48
NMS	CenSurE	0.00	0.79	1.00	0.08
NMS	MSD	0.00	0.79	1.04	5.38
NMS	H.L.	0.08	0.77	1.18	0.62
ANMS	MSER	0.00	0.72	1.15	3.91
NMS	BRISK	0.01	0.74	1.44	1.08
NMS	GFTT	0.04	0.73	1.00	0.06
NMS	AGAST	0.00	0.68	2.08	0.14
NMS	FAST	0.01	0.67	1.01	0.04

Texture dependent keypoint repeatability performance is presented in Figure 4.2. Here, we observe that all evaluated detectors have their lowest keypoint repeatability for images of wooden texture, in some cases with a significant difference to the texture with second lowest repeatability. Among the evaluated keypoint detectors, SIFT has the most stable performance for all textures.

For the transformation dependent repeatability performance, presented in

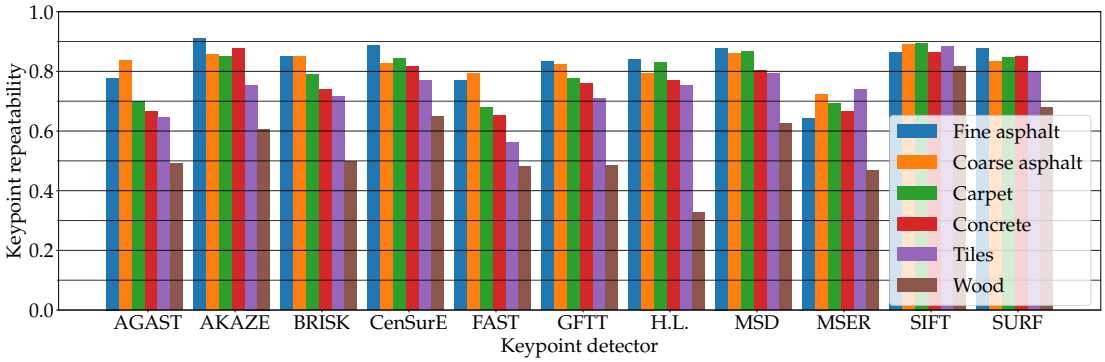


Figure 4.2: Keypoint repeatability of varying keypoint detectors for different types of ground textures from the Micro-GPS database [Zhang et al., 2019] (fine and coarse asphalt, carpet, concrete, tiles, and wood), averaged over all types of evaluated synthetic transformations (rotation, translation, noise, and illumination changes). This figure is adapted from [Schmid et al., 2019, supplementary].

Figure 4.3, we find that all keypoint detectors, but SIFT, are mostly unaffected by the synthetic translation, while the synthetic noise is the most difficult transformation to deal with for all keypoint detectors. It becomes clear that SIFT, AKAZE, and SURF have the best repeatability performance due to them being less affected by the synthetic noise.

Overall, our evaluation suggests that SURF and CenSurE, as well as AKAZE for non-wood texture, are the best detectors on ground texture images. They have among the best repeatability, and ambiguity scores, and are, unlike SIFT and MSD, fast to compute. Still, SIFT can be considered as well, due to its stable performance among all evaluated textures and transformations.

4.4.2 Evaluation of Detector-Descriptor Pairings

We evaluate the pose estimation success rate for all working detector-descriptor pairs. Some pairings are not feasible. The AKAZE description method only allows to use AKAZE keypoint objects. DAISY requires keypoint objects to specify orientation. The ORB description method has requirements on the keypoint object scaling, which excludes SIFT and LIFT.

We evaluate on image pairs from incremental localization, presenting averaged results in Table 4.4. The corresponding results for absolute localization are presented in Table 4.5. The intersection of the sequentially taken image pairs is on average 22.7% with a maximum of 50.0%. Some almost non-overlapping pairs with intersections as low as 1.7% are particularly challenging. The image pairs from the absolute localization tasks have larger intersections with an average of

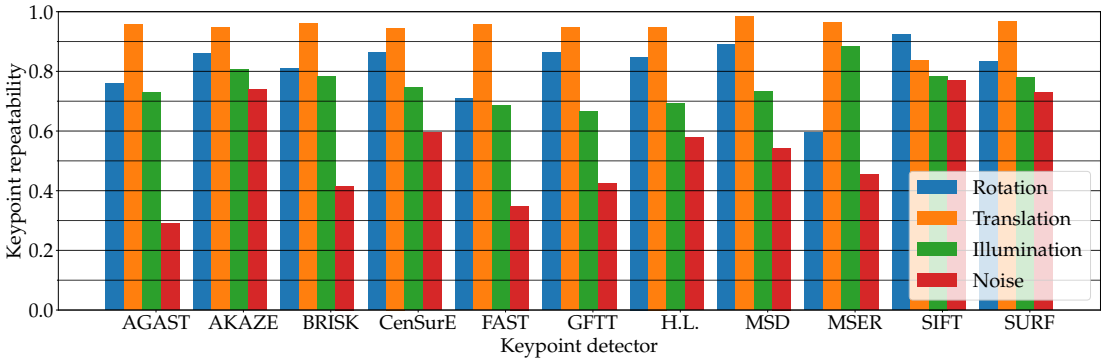


Figure 4.3: Keypoint repeatability of varying keypoint detectors for the individual evaluated synthetic transformations (rotation, translation, noise, and illumination changes), averaged over all textures of the Micro-GPS database [Zhang et al., 2019] (fine and coarse asphalt, carpet, concrete, tiles, and wood). This figure is adapted from [Schmid et al., 2019, supplementary].

Table 4.4: Pose estimation success rates evaluated for incremental localization tasks, where the pose of one image is computed relative to that of a consecutive image recording. For each feature description method, we highlight in bold the respective keypoint detector pairings with largest resulting success rates.

Detector	Descriptor									
	ORB	BRIEF	LATCH	SURF	SIFT	AKAZE	LIFT	BRISK	FREAK	DAISY
CenSurE	**0.93**	**0.90**	**0.90**	0.24	0.82	NA	**0.84**	0.80	0.70	NA
MSD	**0.93**	**0.90**	**0.90**	0.68	0.83	NA	**0.84**	**0.81**	0.69	NA
H.L.	0.88	0.86	0.82	0.75	0.60	NA	0.62	0.74	0.67	NA
SURF	0.89	0.86	0.86	**0.87**	0.73	NA	0.40	0.72	0.73	**0.72**
AKAZE	0.80	0.85	0.83	0.47	0.73	**0.84**	0.76	0.80	**0.74**	0.66
FAST	0.89	0.85	0.85	0.24	0.79	NA	0.77	0.76	0.66	NA
GFTT	0.88	0.84	0.84	0.21	0.80	NA	0.80	0.79	0.68	NA
LIFT	NA	0.84	0.83	0.36	0.69	NA	0.75	0.80	0.72	0.66
MSER	0.83	0.83	0.85	0.59	0.76	NA	0.47	0.62	0.56	NA
SIFT	NA	0.82	0.84	0.59	**0.84**	NA	0.61	0.70	0.63	0.69
AGAST	0.77	0.76	0.75	0.30	0.66	NA	0.61	0.64	0.55	NA
ORB	0.70	0.71	0.71	0.73	0.58	NA	0.12	0.49	0.62	0.47
BRISK	0.66	0.69	0.70	0.61	0.61	NA	0.38	0.71	0.66	0.48

43.7%, ranging from 4.6% to 95.3%. However, in this case, the rotation between the images is with an average of 120 ° (taking absolute orientation differences with a range from 0 ° to 180 °) higher as for the pairs from incremental localization with an average rotation of 3 °. Again, the number of retrieved keypoint objects is reduced to 1000 using the respective best selection method. The best performance for incremental localization of 93% success rate is achieved with ORB on CenSurE or MSD keypoint objects. BRIEF and LATCH perform also well with 90% success rate, also using CenSurE or MSD keypoint objects.

Exemplary for the texture-dependent performance, Figure 4.4 presents for incremental localization the success rates of all descriptors with their respective best performing detector pairing. In cases of multiple best performing detectors, we consider the faster one as better. We observe that all of the best descriptor-detector pairings achieve close to perfect success rates on the fine asphalt and carpet datasets. The wood dataset presents itself to be particularly challenging for several pairings like SIFT-SIFT, MSD-BRISK, SURF-DAISY, and AKAZE-FREAK.

For absolute localization, most feature description methods can achieve more than 90% success rate if paired with the right detector. SURF and DAISY are not quite as successful as they struggle again with images of wooden texture with success rates of their best performing pairings of 0.25 for SURF-SURF and

Table 4.5: Pose estimation success rates evaluated for absolute localization tasks, where the pose of one image is computed relative to that of an independently recorded image, which has been recorded in a separate image sequence. For each feature description method, the respective keypoint detector pairings with largest resulting success rates are highlighted in bold. Description methods marked with a $^+$-symbol can compute keypoint object orientation themselves. The remaining description methods depend on the detector for that information. Detector methods marked with a $^+$-symbol can provide keypoint object orientation to the description method. If a detector method that cannot provide keypoint object orientation is paired with a descriptor method that depends on the detector providing that information, performance is low for absolute localization.

Detector	Descriptor									
	ORB	BRIEF	LATCH	SURF$^+$	SIFT	AKAZE$^+$	LIFT	BRISK$^+$	FREAK$^+$	DAISY$^+$
CenSurE	0.14	0.11	0.09	0.50	0.10	NA	0.11	0.88	0.96	NA
MSD	0.14	0.11	0.09	0.72	0.09	NA	0.10	0.86	0.90	NA
H.L.	0.14	0.10	0.09	0.71	0.10	NA	0.10	0.87	0.89	NA
SURF$^+$	0.89	0.98	0.98	**0.87**	0.94	NA	0.91	**0.96**	**0.98**	0.70
AKAZE$^+$	**0.94**	**0.99**	**0.99**	0.73	0.96	**0.99**	**0.96**	0.92	**0.98**	**0.75**
FAST	0.14	0.11	0.09	0.46	0.10	NA	0.11	0.84	0.89	NA
GFTT	0.14	0.11	0.09	0.37	0.10	NA	0.11	0.86	0.89	NA
LIFT$^+$	NA	0.71	0.85	0.17	0.70	NA	0.52	0.46	0.73	0.24
MSER	0.13	0.10	0.09	0.76	0.09	NA	0.10	0.90	0.89	NA
SIFT$^+$	NA	0.97	**0.99**	0.75	**0.99**	NA	**0.96**	0.93	0.94	0.74
AGAST	0.12	0.10	0.09	0.46	0.08	NA	0.10	0.80	0.86	NA
ORB$^+$	0.82	0.86	0.89	0.84	0.90	NA	0.73	0.84	0.87	0.56
BRISK$^+$	0.79	0.88	0.89	0.76	0.84	NA	0.84	0.87	0.89	0.53

0.07 for AKAZE-DAISY. Detectors that provide orientation information (SIFT, SURF, AKAZE, ORB, BRISK, and LIFT) outperform the other detectors. This is particularly the case for the description methods ORB, BRIEF, LATCH, SIFT, and LIFT which do not compute orientation themselves but rely on it to be readily available. In these cases, if orientation information is not available, pose estimation success rate drops to about 10% to 15%.

For further analysis of feature description performance, we use synthetic transformations to evaluate the pairings of methods for detection and description that performed the best on absolute localization. Again, for each description method we consider its best performing detector pairing (based on the rounded success rates shown in Table 4.5), using the faster one in cases of multiple best performing ones. Additionally, for comparison, we provide the results of BRIEF on CenSurE keypoint objects, which is one of the best performing pairings for incremetal localization tasks, with CenSurE being one of the detectors that does not provide orientation information and BRIEF being one of the description methods that requires that information from the detector. Table 4.6 presents results for feature matching and pose estimation. Here, the use of synthetic transformations allows us to accurately evaluate classification precision and the number of correct matches. We note that the challenges of our synthetic transformations, which include severe rotations and photometric modifications, are more similar to the ones of absolute localization. Consequentially, BRIEF has significantly better performance using AKAZE keypoint objects instead of CenSurE keypoint objects, which lack orientation information, but were well suited for incremental localization. SIFT-SIFT outperforms the other feature

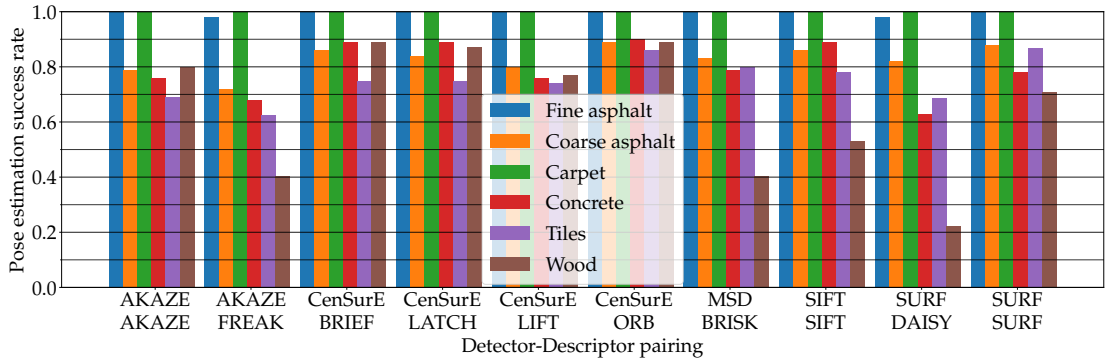


Figure 4.4: Texture dependent pose estimation success rates for incremental localization tasks, where the pose of one image is computed relative to that of a consecutive image recording. We present the results for each description method with its best performing detector pairing on the incremental localization task. This figure is adapted from [Schmid et al., 2019, supplementary].

Table 4.6: Evaluation of detector-descriptor pairings on synthetically transformed images, presenting similar challenges to those occurring for absolute localization. We evaluate the pose estimation success rate, the mean number of matches that are considered to be correct, and the precision metric. The presented values are the averages over all synthetic transformations. Each description method is evaluated with its best performing detector pairing, according to our results on the absolute localization task. Additionally, we evaluate the pairing of CenSurE with BRIEF as one of the best performing pairings on the incremental localization task. Note that neither CenSurE nor BRIEF can compute keypoint object orientation.

Detector	Descriptor	Success rate	#Correct matches	Precision
SIFT	SIFT	1.00	551	1.00
SURF	BRISK	0.99	559	1.00
AKAZE	AKAZE	0.98	545	0.99
AKAZE	BRIEF	0.98	534	0.99
AKAZE	FREAK	0.98	561	0.99
AKAZE	LATCH	0.97	538	0.98
SURF	SURF	0.97	509	0.99
AKAZE	DAISY	0.87	410	0.98
CenSurE	BRIEF	0.77	460	0.92

extraction pipelines. Also, we find precision to correlate with the pose estimation success rate, while this relation is not that clear for the number of correct matches. For example, BRIEF on CenSurE has about 50 more correct matches than DAISY on AKAZE, despite having a significantly lower pose estimation success rate.

In another experiment, we examine the pose estimation performance for incrementally recorded image pairs using different numbers of reference image features, while the maximum number of extracted query image features is kept constant at 1000. Results are presented in Table 4.7. Here, we find our selection of $n = 100$ for the $< n$ KPs performance indicator is validated, as localization performance tends to be low for 100 or less reference image keypoint objects. On the other hand, pairings like CenSurE-ORB, CenSurE-LATCH, and SIFT-SIFT reach values close to their best performance at 300 features already. Others, like SURF-SURF and SURF-DAISY should not be used with less than 500 features per image.

Furthermore, assuming that the number of RANSAC inliers in successful localization attempts correlates with the number of correct matches, we find further evidence for our observation that the number of correct matches is not a suitable indicator for localization performance. This is as we observe that the

Table 4.7: Pose estimation success rates for incremental localization tasks, depending on the maximum number of considered features from the reference images. Each description method has been paired with its best performing detector, when considering 1000 features per reference image.

Detector	Descriptor	10	50	100	200	300	500	750	1000	1500
CenSurE	ORB	0.01	0.54	0.75	0.87	0.89	0.91	0.92	0.92	0.92
CenSurE	BRIEF	0.00	0.47	0.73	0.84	0.87	0.88	0.89	0.89	0.90
CenSurE	LATCH	0.13	0.65	0.77	0.84	0.86	0.88	0.88	0.89	0.89
SURF	SURF	0.01	0.25	0.36	0.56	0.69	0.83	0.84	0.86	0.86
SIFT	SIFT	0.12	0.56	0.71	0.80	0.82	0.83	0.84	0.83	0.84
CenSurE	LIFT	0.04	0.50	0.67	0.76	0.80	0.82	0.82	0.83	0.83
AKAZE	AKAZE	0.03	0.41	0.62	0.73	0.77	0.81	0.83	0.83	0.83
MSD	BRISK	0.03	0.28	0.46	0.69	0.74	0.79	0.81	0.80	0.80
AKAZE	FREAK	0.01	0.33	0.46	0.59	0.65	0.71	0.72	0.73	0.74
SURF	DAISY	0.00	0.16	0.19	0.31	0.45	0.65	0.70	0.71	0.71

number of RANSAC inliers in successful localization attempts keeps increasing together with the number of extracted reference image features, while the localization success rates stagnate at some point. For CenSurE-BRIEF, for example, the number of inliers in successful localization attempts increases from about 35 at 300 reference image features to 94 for 1500 reference image features, even though success rate increases only from 87% to 90%. This suggests that once a certain number of correct matches is available, localization performance does not increase further.

4.5 Discussion

We examined keypoint detectors, selection methods, and feature descriptors on synthetically transformed ground images as well as on pairs of separately taken ground images.

In contrast to Otsu et al. [2013], we find with SIFT a keypoint detector that performs well on all evaluated ground types. For image pairs, where the transformation consists mainly of a translation, as it is the case for the task of incremental localization, we can confirm the suitability of ORB descriptors on CenSurE keypoint objects as well as SIFT features, and the weaknesses of FAST and GFTT keypoint objects, and BRISK and FREAK descriptors, as assessed by Kozak and Alban [2016]. This is even though our evaluation has shown that their metric, the number of correctly matched features, is not necessarily a

good indicator for localization performance. However, in contrast to Kozak and Alban [2016], we observed good performance of SURF. Finally, we validated the observation of Zhang et al. [2019] that SIFT is suited for absolute localization as it is among the best performing methods for the estimation of transformations between image pairs that are recorded at different times and different poses, and the best feature extractor to deal with even more severe synthetic transformations. However, other pairings like BRIEF, LATCH, and AKAZE descriptors on AKAZE keypoint objects perform similarly well and are significantly faster to compute. Overall, we can recommend using ORB, BRIEF, or LATCH on CenSurE keypoint objects for incremental localization and SIFT for absolute localization if sufficient computation power is available. We found classical feature extractors to outperform the evaluated deep learning approach (LIFT [Yi et al., 2016]). However, performance of deep learning approaches could be improved with domain specific training as shown in [Zhang et al., 2019].

This survey allows us to find approaches to feature extraction and their respective parameter settings that are generally suitable for ground images. However, we examined a simplified problem in which only a single reference image is considered to find correspondences with the query image. In the following chapters, we will consider more difficult localization problems with large sets of available reference images and we will consider localization pipelines as a whole, which, for example, use different techniques for feature matching.

5 Identity Matching with Compact Binary Descriptors

Contents of this chapter were partially published in [Schmid, Simon, and Mester, 2020a] and [Schmid, Simon, Radhakrishnan, Frintrop, and Mester, 2022].

Our goal in this chapter is to develop a self-contained ground texture based localization approach to solve the Problems 2 (map creation) and 3 (localization), i. e. an approach to the map creation and localization problems that is sufficient as the single source of localization information as it can be used for the initial global localization as well as for subsequent local localization updates.

Previous approaches require an initial localization estimation from an external source [Kelly et al., 2007, Fang et al., 2009, Nagai and Watanabe, 2015, Kozak and Alban, 2016], making them unsuitable for a self-contained localization system, or they are slow to compute for incremental localization updates [Zhang et al., 2019, Chen et al., 2018], which limits the achievable localization accuracy. For example, if a warehouse robot with a typical velocity of 10 km/h has a localization latency of 200 ms, the robot moves more than 0.5 m during a localization update. The path taken during a localization update can be estimated based on the robot odometry, but without an absolute reference, the resulting relative localization estimate is subject to drift.

We present an adaptation to the approach of Zhang et al. [2019] that performs fast localization updates as it is able to focus on a restricted area of the map according to a prior pose estimate. Our method employs compact LATCH [Levi and Hassner, 2016] descriptors with less than two bytes per descriptor. Also, we introduce *identity matching*, where only identical descriptors are considered as matches, and use it as a substitution of approximate nearest neighbor search. These changes allow us to scale the computational effort of localization according to the confidence in a prior pose estimate, while increasing the localization success rate compared to global methods that do not take advantage of such a prior. Furthermore, we contribute the first quantitative evaluation of ground texture based localization approaches. We compare our approach

to Micro-GPS [Zhang et al., 2019], a global method, Ranger [Kozak and Alban, 2016], a local method, and StreetMap [Chen et al., 2018], which can be used for both tasks.

The related work of existing approaches to ground texture based localization has been introduced in Section 2.4. Our localization method, the *GTBL Method*, is introduced in Section 5.1. Subsequently, Section 5.2.1 presents the experimental evaluation on the Micro-GPS database, including a detailed description of our implementations of the examined localization methods. Additional experimental evaluations on the HD Ground Database [Schmid et al., 2022] are presented in Section 5.2.2, and, lastly, we discuss our insights from the work presented in this chapter in Section 5.3.

5.1 GTBL Method

We adapt Micro-GPS (see Section 2.4.1.1), the localization pipeline of Zhang et al. [2019]. The authors show that Micro-GPS achieves reliable high-precision localization on most of the evaluated ground textures, but it requires more than hundred milliseconds for each localization request, even on a fast computer with a dedicated graphics processing unit.

We identify the construction of a global ANN search structure for feature matching, as a major drawback of Micro-GPS. It allows to perform efficient feature matching between query and reference images; however, the structure represents a fixed set of reference images and needs to be recomputed whenever another image is added to the map. Updating a reference image with a more recent recording requires recomputation as well. Also, using this matching technique means that by default correspondences are always searched globally, considering all reference images of the map, because the method cannot use a localization prior to reduce the number of considered reference images to those that are close to the given position estimate.

We tackle these drawbacks, using *identity matching* in conjunction with compact binary feature descriptors.

For feature extraction, we determine keypoint objects and their orientations using SIFT [Lowe, 2004], and compute feature descriptors with LATCH [Levi and Hassner, 2016]. The SIFT feature detector locates regions of interest as local extrema on a Gaussian scale-space pyramid. LATCH computes binary descriptors for keypoint objects through the comparison of image patch triplets. Each patch represents k pixels $\times$ k pixels, with k being a parameter to be chosen

by the user. Levi and Hassner [2016] suggested not to smooth the image patches. However, for our application, we observed better results employing Gaussian blur as described in the implementation details following later. An anchor patch p_a, is extracted at the position of a keypoint object and is then compared to two surrounding image patches p_1, p_2 using the Frobenius norm. Each bit value of the LATCH descriptor is evaluated by one triplet, each of which specifies a unique placement of p_1 and p_2 with respect to the anchor patch p_a that remains centered at the keypoint object position. A triplet is evaluated to 1 if p_a is more similar to p_2 than to p_1 and to 0 otherwise. We take advantage of the original LATCH triplet arrangements, which have been optimized by the authors. The order of the employed triplets is a ranking based on how many times a triplet has the same value for corresponding keypoint objects and different values for non-corresponding ones. Furthermore, strongly correlating triplets were removed. In our case, we use only the concatenation of the binary responses of the first 15 triplets as a compact binary descriptor. This number results in the highest success rate for our number of extracted features (850). A higher number of bits increases the inlier-to-outlier ratio, but decreases the absolute number of inliers, i. e. the number of correctly matched features. To compensate for this, we would have to extract more features, increasing computation cost and memory consumption.

Our matching strategy proposes only those pairs of descriptors as matches that have identical values. Identity matching can be implemented efficiently as table lookup, i. e. row i of the table contains references to the features with descriptors whose numerical representation of their binary string is equal to i. For feature matching of binary descriptors with a dimensionality of n ($n = 15$ in our case), the lookup table has a length of 2^n. The lookup table is created for a set of reference image features. Then, to find matches for a query image feature, it is sufficient to retrieve the reference features of the table row that corresponds to the query image feature descriptor.

With identity matching, in contrast to the ANN search index employed by Zhang et al., it is not necessary to compute one search structure for the entire map, but feature matching can be performed on an image to image basis. Accordingly, during mapping, we create a descriptor table for each reference image. If a localization prior $[\mathbf{R}_\mathrm{p}|\boldsymbol{t}_\mathrm{p}]_\mathrm{q}^\mathrm{M}$ is available, only the tables of the closest reference images are considered for feature matching, e. g. all reference images with a maximum spatial distance of d_p to the prior. For global localization without prior ($d_\mathrm{p} = \infty$), all tables are considered.

The use of identity matching with compact binary descriptors leads to a large

number of incorrectly proposed matches (outliers). For example, on the Micro-GPS database, which contains roughly 2000 to 4000 reference images per texture, we observe for global localization (considering all reference images during the localization attempt) that typically less than 0.015% of matches can be considered correct correspondences (considering RANSAC inliers of successful localization attempts as correct correspondences). This is why we employ the voting procedure of Micro-GPS [Zhang et al., 2019] for outlier rejection (see Section 2.3). Here, the outlier matches distribute their votes for the current camera position equally on the voting map, while the inlier votes are concentrated in a narrow region (see Figure 5.2 on page 72). Subsequently, the matches that voted for the map cell with most votes are used for a RANSAC-based estimation of the camera pose.

5.2 Evaluation

We evaluate the proposed GTBL Method and state-of-the-art localization methods on the Micro-GPS database in Section 5.2.1 and on our HD Ground Database in Section 5.2.2.

The implementations of the evaluated methods, respectively their parametrization, are slightly different for the two databases and are therefore explained in the corresponding sections.

For both databases, we separately evaluate localization methods for initial localization without available prior and for subsequent local localization with available prior, i. e. an approximate pose estimate is provided as input to the localization method. Our main performance metric is the pose estimation success rate, i. e. the proportion of localization queries for which the estimated pose $[\mathbf{R}_{\text{est}}|\boldsymbol{t}_{\text{est}}]_{\text{q}}^{\text{M}}$ is closer to the actual pose $[\mathbf{R}|\boldsymbol{t}]_{\text{q}}^{\text{M}}$ than d_{t} with an absolute angle difference of less than o_{t}. Here, we adopt the thresholds of Zhang et al. [2019] of $d_{\text{t}} = 4.8\,\text{mm}$ and $o_{\text{t}} = 1.5\,^{\circ}$.

For all our evaluations, the employed hardware consists of an E3-1270 Intel Xeon CPU at 3.8 GHz, and a Quadro P2000 Nvidia graphics card (used to compute SIFT features in Micro-GPS).

5.2.1 Evaluation on the Micro-GPS Database

This section presents the evaluation published in [Schmid et al., 2020a], where we use the six texture types of the ground texture image database of Zhang et al.

[2019] recorded with a PointGrey CM3 camera, described in Section 3.2.

Prior to the evaluation, we find suitable parameters of the examined methods, if not specified by the respective authors, on a training set of 100 query images per ground texture type, optimizing for pose estimation success rate first and for computation time second. Subsequently, separate sets of 500 images per texture type are used to evaluate our experimental setups.

Evaluation of Global Localization Besides our method, we evaluate Micro-GPS [Zhang et al., 2019], for which the code is provided by the authors, and StreetMap [Chen et al., 2018], which is re-implemented according to the paper.

Evaluation of Local Localization For our examination of localization performance with available localization prior, we evaluate our method, StreetMap, and Ranger [Kozak and Alban, 2016], which we re-implemented according to the system description of the authors.

Here, we evaluate pose estimation success rates for varying accuracies d_p of the localization prior. The prior is generated by taking the ground truth pose of the query image, which is then in a first step translated with a specified distance d ($d_\mathrm{p} = d$) into a randomly sampled direction, and in a second step it is rotated with an orientation angle sampled from a zero-mean normal distribution.

All of the local localization methods evaluated in this chapter use the prior only to select a subset of closest reference images to the current pose estimate. If this subset of considered closest reference images is too small, the reference images actually overlapping with the query image might not be included, making correct localization impossible. Therefore, dependent on the available prior accuracy d_p, we empirically determine sufficiently large numbers of considered reference images, ensuring that the reference images actually overlapping with the query image are included.

5.2.1.1 Implementation

In the following, we present implementation details of the evaluated localization methods. We describe the respective function for image processing $\mathrm{f_q}$, map creation m in the sense of Problem 2, and localization g in the sense of Problem 3. For all of the evaluated methods, feature extraction is the same for reference and query images ($\mathrm{f_r}$ is equal to $\mathrm{f_q}$).

GTBL Method

Image processing: We employ the SIFT [Lowe, 2004] implementation of the OpenCV 4.0 library [Bradski, 2000] to extract keypoint objects. The number of layers per pyramid octave is set to 11, the contrast threshold to 0.005, the edge threshold to 13, and the sigma of the employed Gaussian filter is set to 8.5. Only the 850 keypoint objects with largest response values are kept. Then, we extract for each keypoint object the first 15 bit of the LATCH descriptors.

In order to deal with varying image orientations, we use the LATCH variant that rotates the considered image patch according to the keypoint object orientation. The half-size of the evaluated patches is set to 8, making the patches 17 pixels $\times$ 17 pixels, and the sigma of the employed Gaussian smoothing is set to 2.2.

Mapping: For each reference image, the identity matching table is built. These tables are sparsely populated, which is why we implement them as dictionaries that map feature descriptors to lists of indexes from features with that descriptors. To use available priors, a k-dimensional tree (k-d tree) is constructed from the pose estimates of the reference images, using the nanoflann library [Blanco and Rai, 2014].

Localization: If a localization prior is available, only the closest reference images are considered. Otherwise, we perform identity matching with all reference images. The retrieved matches are used to cast votes for the corresponding camera positions on a voting map. The cell size of the voting map grid is set to 75 pixels $\times$ 75 pixels (12 mm $\times$ 12 mm). We select the matches that voted for the voting map cell with most votes, i. e. the voting peak, and perform RANSAC-based pose estimation with them.

Micro-GPS (code is provided by the authors)

Image processing: Zhang et al. [2019] use SiftGPU[1] to extract SIFT features. As for all evaluated localization methods, features are extracted from full-scale images. The authors employ PCA dimensionality reduction to reduce the size of the SIFT descriptors. In our case, the PCA basis for that purpose is created using the entire set of reference images of the currently evaluated texture type. We employ 16-dimensional descriptors, which the authors found to perform better than 8-dimensional ones [Zhang et al., 2019].

Mapping: Of each reference image 50 16-dimensional SIFT features are randomly sampled. The authors assume that corresponding features will have

[1] https://github.com/pitzer/SiftGPU

similar scale. Therefore, they use the scale information to divide the set of reference features into 10 groups. For each group, they construct an ANN search index with the FLANN library [Muja and Lowe, 2009].

Localization: For each 16-dimensional SIFT feature of the query image, its ANN reference feature is retrieved, using the search index corresponding to the feature's scale. Each of the obtained matches casts a vote for the camera position on a voting map with a cell size of 50 pixels $\times$ 50 pixels (8 mm $\times$ 8 mm). Afterwards, the matches that voted for the voting map cell with most votes are used for RANSAC-based pose estimation.

StreetMap (Without Prior)

Image processing: We extract SURF [Bay et al., 2006] features using OpenCV [Bradski, 2000], using 4 pyramid octaves with 3 layers each, and a Hessian threshold of 20. Per image the 1000 features with largest response values are kept for further processing.

Mapping: For each image, a BoW representation is computed based on the retrieved SURF features, using the FBOW library [Muñoz-Salinas and Medina-Carnicer, 2020]. The vocabulary for that purpose was computed beforehand, using default parameters of the library and the extracted SURF features of 1000 images per texture type.

Localization: The number of considered reference images is reduced by 80%, by selecting the most similar ones to the query image based on their BoW representations. This value is a trade-off between localization performance and computation time. For matching, we find for each query image feature the most similar reference feature from the remaining reference images, using the L2 norm and the OpenCV brute force feature descriptor matcher. A ratio test with a factor of 0.9 is employed for outlier rejection. Poses are estimated in a RANSAC fashion, using the remaining feature matches.

StreetMap (With Prior)

Image processing: OpenCV [Bradski, 2000] SURF features are extracted from an image pyramid with 5 octaves with 4 layers each. The Hessian threshold for keypoint rejection is set to 20, and only the 768 features with largest responses are kept.

Mapping: A k-d tree [Blanco and Rai, 2014] is built from the positions of the reference images.

Localization: The procedure is the same as for global localization, but the considered reference images are selected based on closeness to the prior, using the k-d tree.

Ranger

Image processing: Kozak and Alban [2016] use CenSurE [Agrawal et al., 2008] keypoint objects, which are not robust to the image orientation. For street vehicles, robustness to orientation is not required because typically the vehicle orientation is similar during mapping and localization. In our evaluation, however, image orientations during mapping and localization are independent of each other. Therefore, we exchange CenSurE with AKAZE [Alcantarilla et al., 2013] keypoint objects, which among the OpenCV [Bradski, 2000] keypoint detectors achieved the best results for our Ranger implementation. The best parameters we found for AKAZE are a response threshold of 10^{-5}, and a single image pyramid octave with two layers. Up to 1250 keypoint objects with largest response values are kept per image. For feature description, we employ the rotation invariant BRIEF description method variant of OpenCV with its full size of 64 bytes.

Mapping: A k-d tree [Blanco and Rai, 2014] is built from the positions of the reference images.

Localization: Features of query image and the closest reference image are matched using the OpenCV nearest neighbor feature descriptor matcher with Hamming distance. For outlier rejection, a cross-check is performed, i. e. the nearest neighbor condition has to be true in both directions: from query image feature descriptors to reference image feature descriptors and vice versa. The remaining feature matches are used for RANSAC-based pose estimation. If the estimated pose is supported by at least 25 matches, it is used as final output of the method. Otherwise, the procedure of matching and pose estimation is repeated with the next closest reference image, and so on. If the condition is not met by any of the considered reference images, we use the pose estimate with most inliers.

5.2.1.2 Results

We present the results on the Micro-GPS database.

Localization without Available Prior Pose estimation success rates for our experimental setup for global localization are presented in Figure 5.1. We ob-

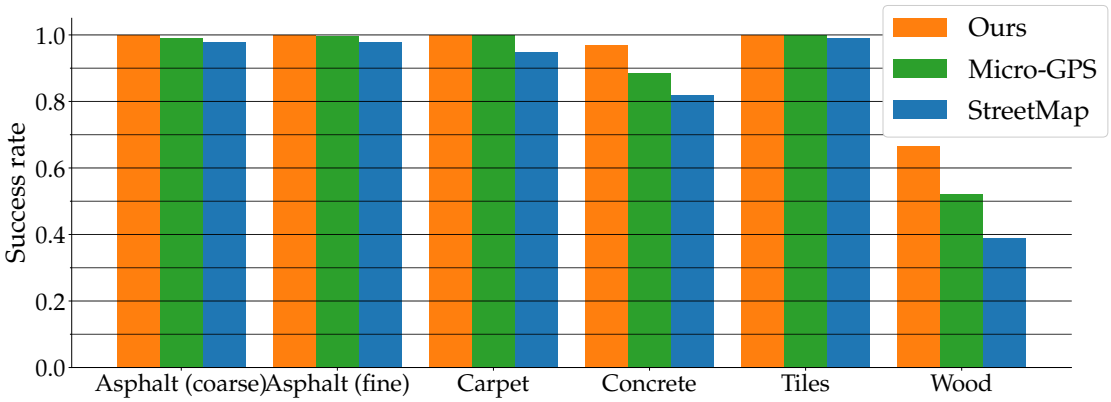


Figure 5.1: Pose estimation success rates for the task of global localization, i. e. without available pose approximation as prior, on the Micro-GPS database [Zhang et al., 2019], evaluated for our GTBL Method, Micro-GPS [Zhang et al., 2019], and StreetMap [Chen et al., 2018]. This figure is adapted from [Schmid et al., 2020a].

serve that both types of asphalt, carpet, and tiles are particularly well suited for ground texture based localization, as all three evaluated methods reach almost perfect success rates. The situation is different for concrete and wood. While our method is still able to localize correctly in 97.0% of the test cases on concrete texture, the original Micro-GPS reaches only 88.4% success rate and StreetMap 82.0%. For wood texture, our method is again the best performing method, but only achieves a success rate of 66.6%, while Micro-GPS and StreetMap have 51.4% and 39.0%, respectively. Further analysis shows that lower success rates can be explained with lower numbers of inliers among the matched features. During localization, our method identifies on average more than 40 inliers for asphalt, carpet and tiles, but only 31.5 for concrete and 9.7 for wood texture images. One explanation for this is a property distinguishing the wood texture from the remaining textures of the Micro-GPS database: it has a fibrous structure. Due to this, the wood texture changes mostly in the traverse direction of the fibers while it changes only little along the fibers. In other words, the wood images have regions of homogeneous texture leading to visual aliasing. As a result, we observe lower keypoint repeatability on wood images. In fact, in Chapter 4, using pairs of synthetically transformed images, we found that wood is the most challenging texture for keypoint detectors to retrieve corresponding keypoint objects.

A voting map is illustrated in Figure 5.2. For better visualization, we doubled the voting cell size to 150 pixels $\times$ 150 pixels (24 mm $\times$ 24 mm). One cell, which is corresponding to the actual camera position, received the most votes, while outlier votes are randomly distributed.

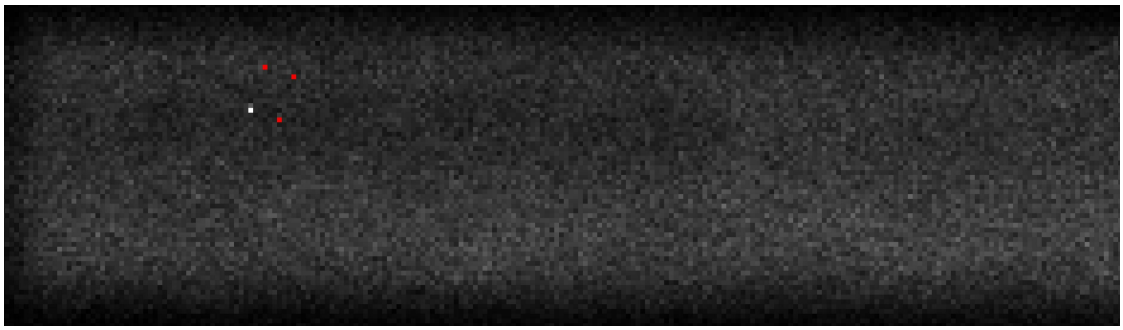
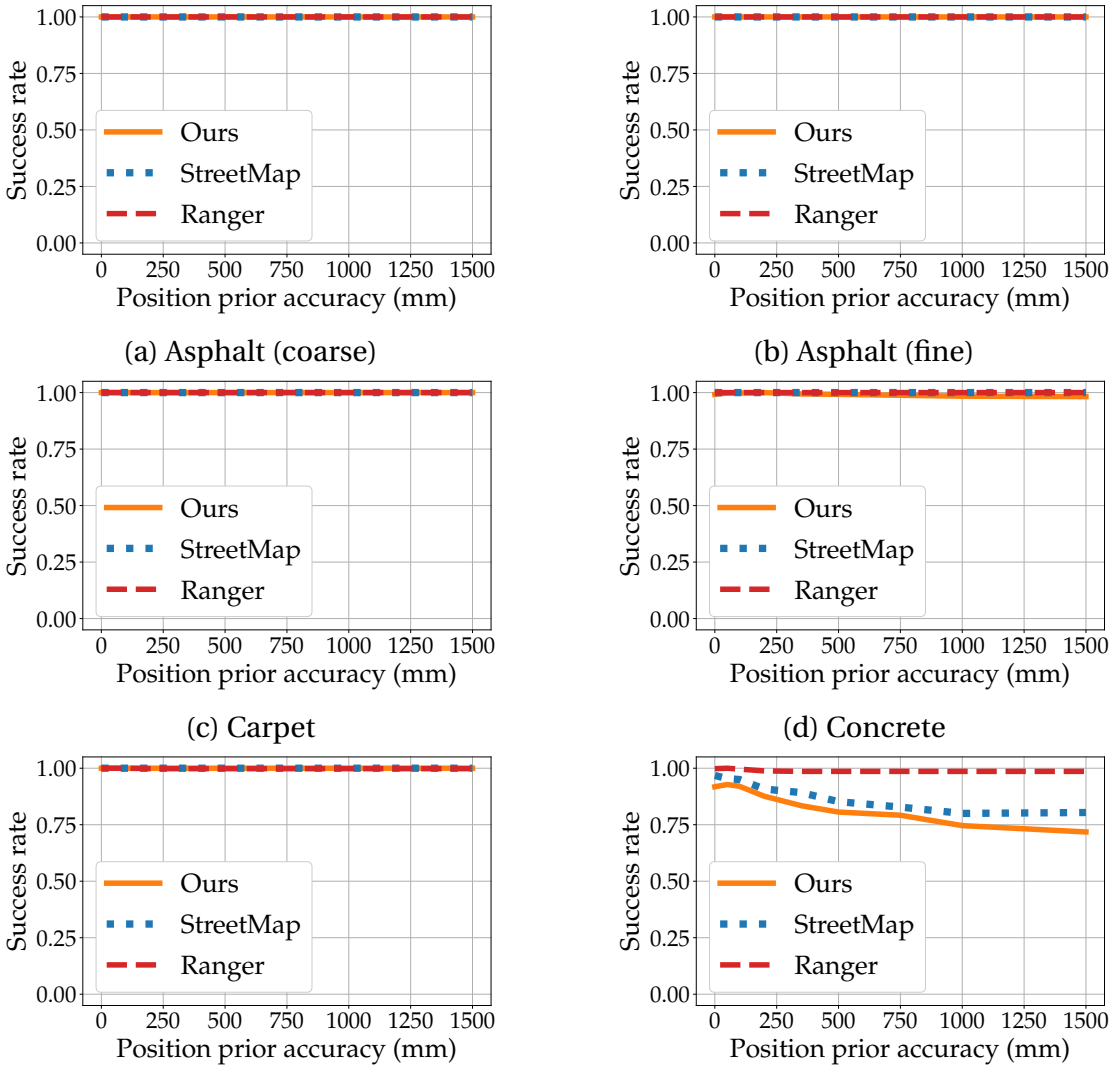


Figure 5.2: Cutout of a voting map from a successful global localization attempt of our GTBL Method. Each pixel in this visualization represents a voting cell that covers an area of about 24 mm × 24 mm in the real world. Pixel brightness indicates the number of feature matches voting for it (see Section 2.3 for a detailed description of the voting procedure of Zhang et al. [2019] that we use for our method). The pixel, representing the voting cell that received most votes, is significantly brighter than the others. Since matches vote for the position of the upper left query image corner in the map coordinate system, we highlight the true positions of the other three corners with red pixels. This allows us to see that the voting procedure was successful in identifying the true position of the top left corner of the query image. This figure is taken from [Schmid et al., 2020a].

Localization with Available Prior For local localization, i. e. localization with available prior, results are presented in Figure 5.3. As explained previously, we empirically determined suitable numbers of reference images that are taken into consideration for a certain prior accuracy. The corresponding fixed numbers can be found in Table 5.1, they are chosen conservatively, i. e. with a tendency to a larger number than it is typically required, to avoid a situation in which localization with the available set of reference images is not possible.

On both asphalt types, carpet, concrete, and tiles, all three evaluated methods are almost always able to localize correctly. Again, wood (Figure 5.3(f)) presents itself as the most challenging ground texture type. With decreasing prior accuracy, localization success rates of StreetMap and our method decline. Again, this can be explained with a low number of inlier matches for wood, which leads to a less significant inlier voting peak than there is for other textures. For increasing numbers of considered reference images, the number of outlier votes increases, and it becomes more likely that variations in the distribution of outlier votes cause higher voting peaks than the peak induced by inliers. Similarly, the inlier-to-outlier ratio of StreetMap decreases with increasing numbers of reference images, while Ranger considers one reference image after the other and is therefore robust to this problem.



(a) Asphalt (coarse)

(b) Asphalt (fine)

(c) Carpet

(d) Concrete

(e) Tiles

(f) Wood

Figure 5.3: Local localization success rates with varying position accuracy. Evaluated on the Micro-GPS database [Zhang et al., 2019] for our GTBL Method, StreetMap [Chen et al., 2018], and Ranger [Kozak and Alban, 2016]. This figure is adapted from [Schmid et al., 2020a].

On wood, our approach is outperformed by both StreetMap and Ranger. However, they become slow for less accurate priors, due to the use of nearest neighbor matching, computing distances between all possible pairings of query feature descriptors and reference feature descriptors. Figure 5.4 presents the required computation time of feature matching for the three evaluated localization methods on the carpet test set. Using a prior with an expected accuracy of 0.35 m, it takes 0.19 s to match features for StreetMap and 0.26 s for Ranger, while our method takes only 0.01 s. If the expected prior accuracy is 1.5 m, feature matching for StreetMap takes 1.87 s and 2.72 s for Ranger, but only 0.11 s for our method, utilizing identity matching of compact binary descriptors.

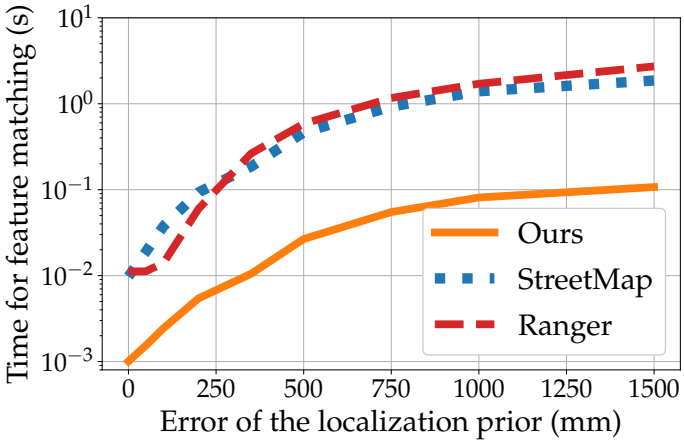


Figure 5.4: Required computation time for feature matching on the carpet dataset of the Micro-GPS database [Zhang et al., 2019] for varying position prior accuracies, evaluated for our GTBL Method, StreetMap [Chen et al., 2018], and Ranger [Kozak and Alban, 2016]. This figure is adapted from [Schmid et al., 2020a].

Table 5.1 presents for our method and Micro-GPS the localization time, without the required time for feature extraction. The computational effort for feature extraction is comparable for both methods, as it is dominated by the use of SIFT. Using SiftGPU, feature extraction takes us about 40 ms. The computational effort of our matching method grows linearly with the number of considered reference images; for large numbers, it is slower than ANN matching approaches. Accordingly, Micro-GPS performs global localization faster than our method. However, in practice global localization is typically performed only in a few and not time-critical cases whenever the kidnapped robot problem occurs. Afterwards, the previous pose estimation can be used as prior for the next localization step. With increasing accuracy of the available prior, less reference images have to be considered, reducing the localization time of our method. If the prior is reliably more accurate than 1.5 m, our method will be faster than Micro-GPS. At the same time, as seen in Figure 5.3, the chance of correct localization increases when using a prior.

The memory consumption of our method is about three and a half times as large as that of Micro-GPS. We roughly estimate the memory requirements as follows. Per reference image, Micro-GPS stores 50 keypoint objects using 4 floating point values for x- and y position, scale, and orientation; additionally, it stores a 16-dimensional floating point descriptor for each keypoint object. If we use 32 bit per floating point value, this results in a total required memory of $(50 \cdot 4 \cdot 32 + 50 \cdot 16 \cdot 32)$ bit $=$ 32000 bit. Our method stores per reference image 850 keypoint objects (with position and orientation), and a dictionary

Table 5.1: Required time for localization *without* the time used for feature extraction, evaluated on the carpet dataset of the Micro-GPS database [Zhang et al., 2019] dependent on the availability and accuracy of the position prior. We evaluate our GTBL Method and Micro-GPS [Zhang et al., 2019], for both of which the time of feature extraction is dominated by the use of SIFT [Lowe, 2004].

Position prior accuracy (mm)	Number of considered reference images	Computation time (ms) Ours	Micro-GPS
0	5	1.60	
50	10	2.42	
100	20	3.85	
200	50	8.12	
350	100	15.25	
500	250	36.74	
750	500	73.87	
1000	750	108.48	
1500	1000	143.65	
∞ (no prior)	2014	286.47	145.55

with 850 pairs of 15-bit descriptors and integer feature indexes, resulting in $(850 \cdot 3 \cdot 32 + 850 \cdot (15 + 16))$ bit $= 107950$ bit.

5.2.2 Evaluation on the HD Ground Database

We present additional evaluations from [Schmid et al., 2022] of the GTBL Method on our HD Ground Database. Section 5.2.2.2 presents the evaluation of global localization, where we examine the GTBL Method and StreetMap [Chen et al., 2018] with the employment of BoW image retrieval. In this case, we use SIFT features as basis for the BoW image retrieval, because we observed better performance as with SURF features (more details on this are presented in Chapter 7). SIFT parameters are optimized on the respective training areas of the HD Ground Database, and the BoW vocabulary is also created with the features extracted on those training areas. Also, we present an experiment that examines the application of our GTBL Method for a teach-and-repeat scenario, which is one of the novel aspects that the HD Ground Database enables us to examine.

In Section 5.2.2.2, we perform a similar evaluation as in Section 5.2.1.2, where a prior pose estimate is available for the localization task. Here, we compare our GTBL Method, Ranger [Kozak and Alban, 2016] and StreetMap [Chen et al., 2018] in its variation that makes use of an available prior. The implementations are similar to those for our evaluation on the Micro-GPS database, but for Ranger, we employ the feature extraction pipeline originally suggested by the authors, using oriented BRIEF descriptors on CenSurE keypoint objects,

instead of AKAZE features as in the previous evaluation. In order to use these features, even though CenSurE keypoint objects are rotation-variant, we use the orientation estimate of the given prior as keypoint object orientation, as suggested in [Schmid et al., 2020b]. The corresponding Standard Deviation (SD) of the orientation prior is set to $3.0\,^{\circ}$. A closer analysis of the implications of this strategy of making use of the orientation prior is done at a later point in Chapter 8.

5.2.2.1 Parameter Configuration

The examined localization methods are adapted to the HD Ground Database using its training areas by repeating two steps: (1) randomly sample a configuration from a pre-defined parameter space, (2) if it has a higher success rate, or a similar success rate but a faster computation time than the previous best, perform a gradient descent like optimization by evaluating configurations with slightly changed values. In contrast to the previous evaluation on the Micro-GPS database, we optimize texture-specifically and we add the scale at which the images are processed as additional free parameter.

Correspondingly, the scale and the number of extracted features per image become two of the most important parameters to optimize. Table 5.2 presents the respective optimized values that we obtained. We observe that for most combinations of texture and localization method, a much lower image resolution would suffice. On carpet, for example, best performance is reached using only 0.20 to 0.35 of our recording resolution of 0.1 mm per pixel. However, in other cases having an image scale of up to 0.88 of our native image scale is beneficial to the success rate.

Other optimized, texture-specific parameter settings are presented in the appendix in Section A.2.

5.2.2.2 Results

We present the results of the described experiments on the HD Ground Database.

Localization without Available Prior Table 5.3 presents the success rates for StreetMap and the GTBL Method. For this evaluation, we use the two regular test sequence that were recorded closest in time to the initial scanning of the application area, i. e. one being approximately one week apart and the second one being approximately two weeks apart from the date of scanning.

Table 5.2: The texture-dependent optimized image scale, indicating the downsizing factor of image resolution compared to the available image resolution of the HD Ground Database [Schmid et al., 2022], and the number of extracted features per image for our GTBL Method, StreetMap [Chen et al., 2018], and Ranger [Kozak and Alban, 2016].

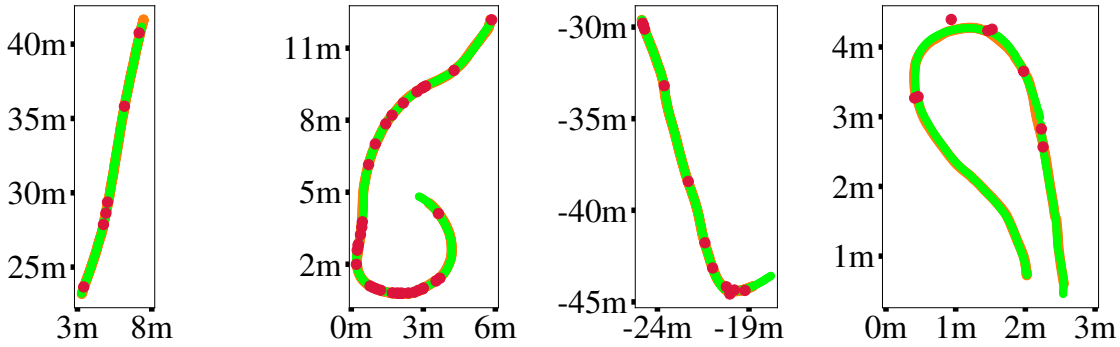
Texture	Approach	Image scale	#Features
Asphalt	Ours	0.60	600
	StreetMap	0.20	200
	Ranger	0.20	400
Cobblestone	Ours	0.34	600
	StreetMap	0.38	400
	Ranger	0.88	650
Carpet	Ours	0.35	600
	StreetMap	0.20	600
	Ranger	0.20	350
Laminate	Ours	0.20	1100
	StreetMap	0.70	900
	Ranger	0.28	350

An analysis of performance depending on the recording date will be done later in Chapter 7. In contrast to our evaluation on the Micro-GPS database, we observe low localization success rates of the GTBL Method, while StreetMap still performs well on cobblestone, carpet, and laminate, but not on asphalt. For further examination of this phenomenon, we evaluate a variant of the GTBL Method that employs BoW image retrieval in the same way StreetMap does, i. e. in both cases we select the same 100 reference images with most similar BoW representation to that of the query image. This variant also achieves better performance on cobblestone, carpet, and laminate. BoW image retrieval improves the localization performance, because less non-overlapping reference images have to be considered by the localization method. A smaller number of considered reference images reduces the probability of experiencing visual aliasing, i. e. another place with similar feature occurrences is confused with the actual place of the agent. However, this improvement through BoW image retrieval does not seem to work well for asphalt, but this can be explained with particularly poor image retrieval performance of the BoW approach for the asphalt recorded in the HD Ground Database, as we will see later on in Chapter 7.

We also examine the use of the GTBL Method (without the use of BoW image retrieval) for global map-based localization in the teach-and-repeat scenario, in which a specific path is recorded multiple times by following a rope. We use one sequence of images recorded while following a certain rope configuration as reference images, and the other (three) sequences following the same rope configuration as query images. Here, localization without prior is less difficult,

Table 5.3: Global localization success rates on the HD Ground Database [Schmid et al., 2022], for the GTBL Method with and without Bag of Words (BoW) image retrieval, and for StreetMap [Chen et al., 2018] with BoW image retrieval.

	Asphalt	Cobblestone	Carpet	Laminate
StreetMap with BoW	0.033	0.401	0.756	0.673
GTBL Method with BoW	0.022	0.333	0.627	0.321
GTBL Method	0.084	0.186	0.042	0.117



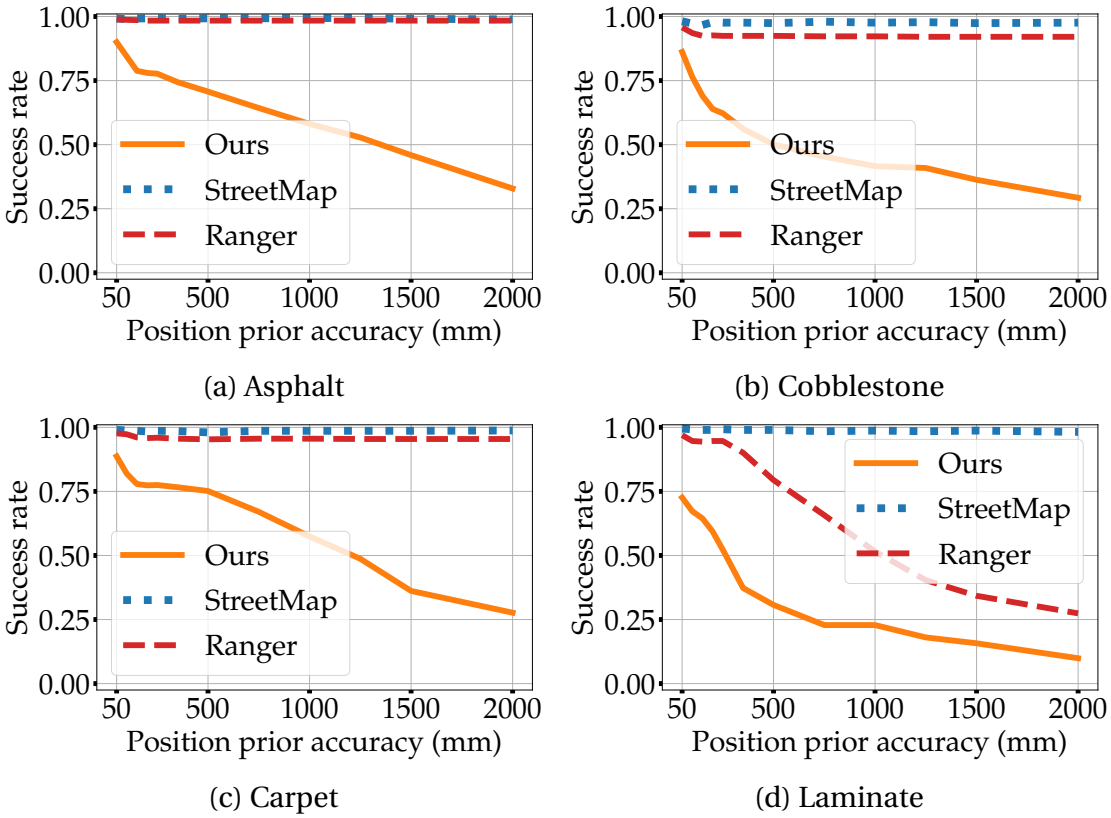
(a) Asphalt (b) Cobblestone (c) Carpet (d) Laminate

Figure 5.5: Evaluation of the GTBL Method for global localization in the teach-and-repeat scenario. Orange dots present the true positions of the path that was driven during the *teach* phase. These are overlaid by the estimated positions of the images from the *repeat* phase. Here, green dots represent successful and red dots represent unsuccessful localization attempts. Positions on the plot axes correspond to the actual metric map coordinates. This figure is taken from [Schmid et al., 2022].

because the sequences are recorded in quick succession, the number of reference images is smaller, and the orientations of test and reference images are either similar or roughly 180 ° rotated. We observe mean success rates of 92.9% on cobblestone, 97.6% on carpet, 92.4% on laminate, and 95.1% on asphalt. Figure 5.5 illustrates for each texture the course and the localization results for one of the recorded teach-and-repeat paths.

Localization with Available Prior If a prior is available, it is sufficient to consider its spatially closest reference images. The radius in which reference images should be considered depends on the confidence in the prior accuracy d_p.

We examine the localization performance of Ranger, StreetMap (without BoW image retrieval), and the GTBL Method. Prior pose estimates are generated in the same way as it was done for the evaluation on the Micro-GPS database, shifting the available ground truth pose with a distance of d and an orienta-



(a) Asphalt

(b) Cobblestone

(c) Carpet

(d) Laminate

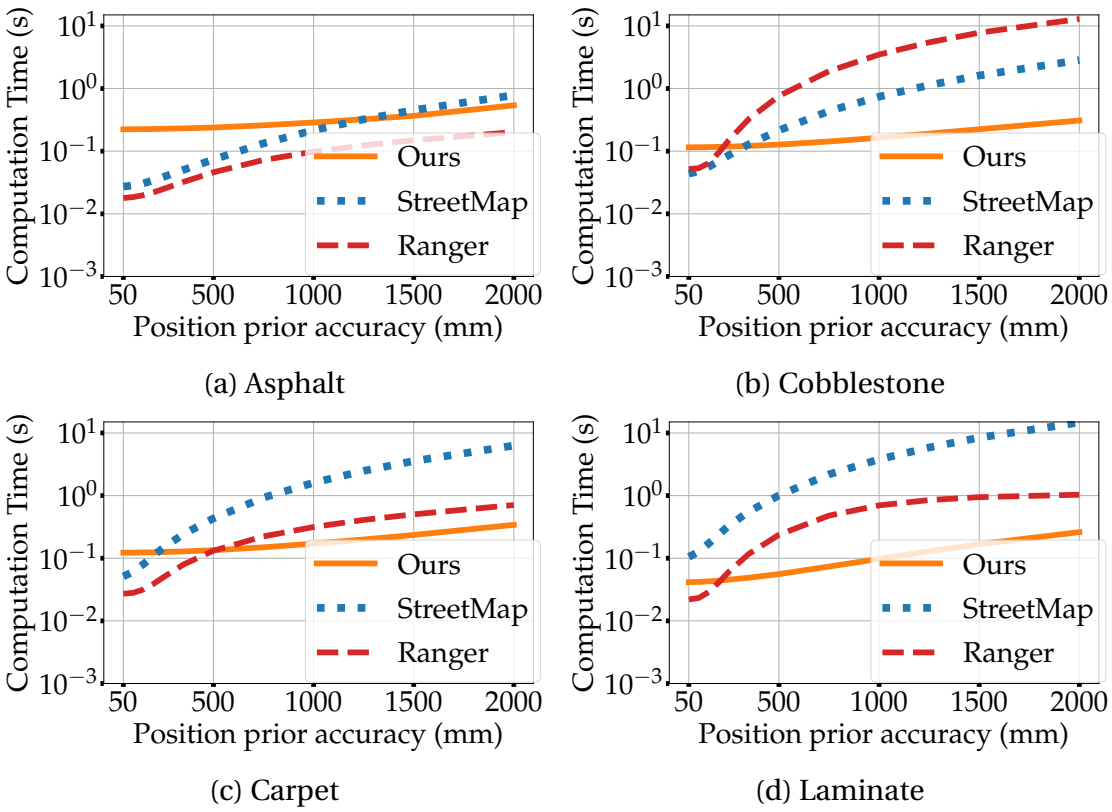
Figure 5.6: Success rates of local localization, i. e. with an approximate pose estimate as prior, on the HD Ground Database [Schmid et al., 2022] for varying position prior accuracies, evaluated for our GTBL Method, StreetMap [Chen et al., 2018], and Ranger [Kozak and Alban, 2016].

tion angle being sampled from a zero-mean normal distribution. Again, for each texture, evaluation is done on the two regular test sequences that were recorded with shortest time distance to the point in time at which the respective application areas were initially scanned.

In our first experiment, the SD of the orientation prior is set to 3.0 °, while we vary the position prior accuracy $d_\mathrm{p} = d$ between 50 and 2000 mm. Depending on d, we adjust the number of considered closest reference images. Let a_P denote the possible area in which we are located, a_I the area covered by an image, and n_inc the number of images we expect each point on the ground to be included in. Then, we compute the number of considered closest images as:

$$\frac{a_\mathrm{P}}{a_\mathrm{I}} \cdot n_\mathrm{inc} = \frac{\pi d^2}{a_\mathrm{I}} \cdot n_\mathrm{inc} = \frac{\pi d^2}{0.12\,\mathrm{m} \cdot 0.16\,\mathrm{m}} \cdot 9. \tag{5.1}$$

Texture-specific results are presented in Figure 5.6. StreetMap achieves very high success rates, independently of the position prior accuracy. The same is true for Ranger, except for laminate with translation distances above 250 mm.



(a) Asphalt

(b) Cobblestone

(c) Carpet

(d) Laminate

Figure 5.7: Required time for local localization (including the time used for feature extraction) for varying position prior accuracies, evaluated for our GTBL Method, StreetMap [Chen et al., 2018], and Ranger [Kozak and Alban, 2016].

The GTBL Method achieves lower success rates and suffers more from inaccurate priors. However, good performance with success rates of at least 0.67 is still reached for position prior accuracies of up to 100 mm.

We also examine the required computation time. Figure 5.7 presents the results. Here, we measure the computation time of the whole localization procedure, including the expensive feature extraction. Accordingly, we observe that for small numbers of considered reference images, the computation time is dominated by the time that is required for feature extraction. This is why, for the employed parametrization, the GTBL Method is the slowest method for accurate position priors. For less accurate position priors, the GTBL Method has an advantage using the identity matching technique instead of brute-force nearest neighbor feature matching, which means that its computation time increases only slightly for less accurate priors.

5.3 Discussion

We propose identity matching, a feature matching strategy based on compact binary feature descriptors, which simplifies feature matching to a single table lookup. Substituting Micro-GPS's [Zhang et al., 2019] use of a global search index for feature matching with our strategy, allowed us to reach higher localization success rates than the state-of-the-art methods for global localization on the Micro-GPS database. On the HD Ground Database, with larger application areas and in particular with significantly larger numbers of reference images, StreetMap achieves better performance, mainly due to the employment of BoW image retrieval to reduce the number of considered reference images. Still, the GTBL Method achieves good performance in the teach-and-repeat scenario, for example.

Furthermore, our method allows to add, remove, and update mapped reference images online without the need of map re-computation. Also, with our matching strategy the method is able to take advantage of prior pose estimates to perform local localization updates. In this case, apart from wood floor texture, our method performs similarly well on the Micro-GPS database as state-of-the-art local localization methods, while being faster to compute, especially for inaccurate prior pose estimates. A similar trend is observed on the HD Ground Database. In particular, for inaccurate priors, which require a larger number of reference images to be considered, the GTBL Method tends to be faster than StreetMap and Ranger. Lower computation times are an advantage, as it can lead to higher effective localization accuracy, with the time between image recording and available pose estimation being shorter, and it enables more frequent pose updates or savings on the required computational power. However, we identify that the gain in matching speed that comes with the employment of identity matching is paid for with a larger memory consumption, due to the larger number of stored features per reference image. Also, on the HD Ground Database, besides the teach-and-repeat scenario, we observe good localization success rates of the GTBL Method only for position prior accuracies of up to 100 mm.

The significant discrepancy in the localization performance of the GTBL Method on the two evaluated databases, especially for global localization without available prior pose estimate, remains an open research question to be examined in future research.

In the remainder of this dissertation, we will examine various ways of improving the GTBL Method. In Chapter 6, we derive a stochastic model that allows to

determine its success rate based on only a few test images of the application area. This model allows us to build an automatic parameter optimization framework that enables efficient texture-specific parametrization of the GTBL Method. Then, in Chapter 7, we propose a method for image retrieval of ground images, similar to BoW that we already examined here, which further improves the global localization performance of our method. Finally, in Chapter 8, we propose a way of avoiding to perform proper keypoint detection, which can significantly reduce the computation time of the GTBL Method for localization with available prior.

6 Model-Based Parameter Optimization

Contents of this chapter were partially published in [Schmid, Simon, and Mester, 2021].

Realizing optimal performance of localization methods typically requires the choice of a variety of parameters, such as the number of considered visual features per image. Finding optimal parameter settings is often a time-consuming process, in which many possible choices are considered. For our GTBL Method, for example, we identify about ten important parameters to be optimized. Even if we would consider only a few possible values per parameter, the total number settings of the method to be evaluated becomes very large. It is therefore desirable to predict the localization performance without having to extensively evaluate the method. This becomes most critical for the time-extensive global localization, where the whole application area is considered for the pose estimation of a given query image.

We introduce a prediction model for the success rate of two state-of-the-art methods for global localization with ground images: Micro-GPS [Zhang et al., 2019], and our GTBL Method [Schmid et al., 2020a]. Our model requires only a few test images of the application area, for each of which we test if the localization methods succeed in estimating its pose in respect to the others. This allows to quickly determine the *local* localization performance. Again, localization attempts are considered to be correct if their position error is below $d_t = 4.8\,\text{mm}$ and the absolute orientation error below $o_t = 1.5\,^\circ$. Assuming similar properties over the application area, the model then uses the local knowledge to estimate the expected *global* localization performance. Based on the predicted global performance, an agent, e. g. a mobile robot, is then able to optimize the localization method according to the challenges of the current ground texture type. In comparison, a simple black-box approach to parameter evaluation would directly evaluate the global localization performance, considering not only a few images, but the entire set of available reference images, which is more accurate but has significantly higher computational cost. Therefore, besides

a deeper understanding of the factors that lead to successful localization of the examined localization methods, our model-based performance evaluation allows to consider more parameter configurations in the same amount of time. Accordingly, the prediction model enables faster deployment of agents in new application areas.

The approach that is proposed in this chapter is the first method for model-based performance evaluation of feature-based localization methods using images of a downward-facing camera.

The structure of the chapter is as follows. After introducing other work on the task of parameter optimization for ground texture based localization methods in Section 6.1, Section 6.2 defines the properties that we expect the examined localization methods to have. Section 6.3 summarizes our prediction model, while the full derivation is presented in the Appendix (Section A.3). Subsequently, Section 6.4 evaluates the predictive power of the model, and we examine its suitability to be used in a parameter optimization framework. Finally, Section 6.5 concludes with a discussion of our findings.

6.1 Related Work

Apart from the success rate, computational effort and memory consumption of the localization method should be optimized. Less computational effort allows to perform localization more quickly, resulting in shorter reaction time of the localizing agent. Alternatively, lower computational effort allows to save electrical energy from the processing or to reduce hardware cost. Optimizing parameter configurations for these goals is a complex task that requires to make trade-off decisions. So far, ground texture based localization methods have been parametrized based on extensive empirical evaluation of possible parameter values [Schmid et al., 2019, 2020a], treating the method as a black-box, without the need of a good understanding about the impact a changed parameter value could have.

An alternative was developed by Mount et al. [2019]. They proposed a method to automatically determine a suitable trade-off between camera coverage area and localization performance, using only a few pairs of aligned test images. Similarly, our approach requires only a few test images of the application area, but it allows to optimize any parameter of feature-based localization methods. Since the camera coverage is not a relevant parameter for the state-of-the-art localization methods being considered in this work, we cannot compare with

Mount et al.'s approach. Also, the approach of Mount et al. [2019] is not model-based, it involves evaluating all considered parameter values on the test images, while our approach can avoid that for some parameters, like the number of extracted features per image.

6.2 Localization Method Properties

We define the properties of the examined ground texture based methods for global localization. The available input consists of a set of reference images $\mathcal{R}$ that have already been properly aligned with each other, and a query image q that we want to localize, which was also recorded in the mapped area.

We consider feature-based methods. The method extracts a set of reference features $\mathcal{F}_\mathcal{R}$ from the reference images (n_r per image), and a set of query features $\mathcal{F}_\mathrm{q}$ from the query image. Extracted keypoint objects specify the orientation of their image patches. Also, keypoint objects of query features specify their position in the query image, while keypoint objects of reference features specify their position in the map. Then, a matching method proposes a set of matches $\mathcal{M}$ as possible correspondences between the feature sets. Every match $m \in \mathcal{M}$ is a pair consisting of a query feature and a reference feature $m = (f_q \in \mathcal{F}_\mathrm{q}, f_r \in \mathcal{F}_\mathcal{R})$. Finally, a pose estimation method uses the matches to estimate the Euclidean transformation $[\mathbf{R}|\boldsymbol{t}]^\mathrm{M}_\mathrm{q}$ projecting the query image q onto the map M.

In addition to the previously described pipeline, we assume that, *prior* to the pose estimation step, examined methods employ the previously introduced *voting procedure* (Section 2.3) for the spatial verification of matches. This allows to reject a proportion of incorrect matches (outliers), which is useful as feature-based global localization methods on ground texture were shown to suffer from large quantities of outliers [Zhang et al., 2019, Schmid et al., 2020a], which can be explained with the fact that individual features are typically not unique for a specific ground region, but rather the spatial composition of multiple features allows for unique identification of a ground region. In this work, we consider a match to be correct, i. e. to be an inlier, if it can be used to determine the correct query image pose (details in Section 6.3.2).

Examined Methods Micro-GPS of Zhang et al. [2019] and the GTBL Method [Schmid et al., 2020a] have the described properties. So, the prediction model, introduced in the following, can be applied to both of them.

6.3 The Prediction Model

A complete derivation of our prediction model for the success rate of a method with the properties described in Section 6.2 is given in the appendix in Section A.3.

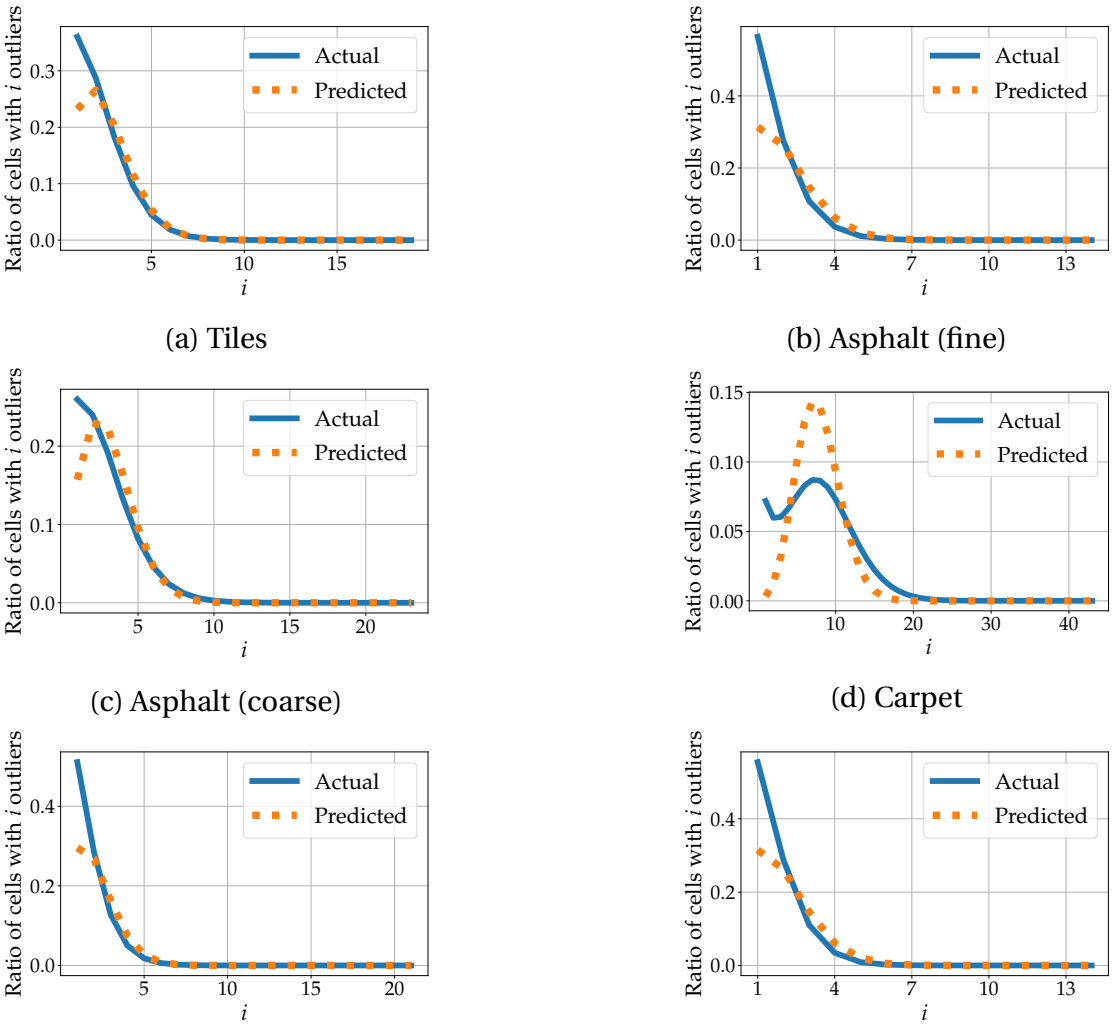
The main assumption of the model is that localization succeeds if among the matches voting for the voting peak v_p are at least two inliers, denoted as $N_{\mathcal{I}}^{v_p} \geq 2$. According to our results, this is an accurate assumption. For both evaluated localization methods, the success rate is greater than 99.3% for localization attempts that hold the condition, while the success rate over all localization attempts is 89.9% for Micro-GPS, and 94.3% for the GTBL Method. Also, not a single localization attempt that does not hold the condition succeeded.

A second important assumption is made about the spatial distribution of outlier votes on the voting map. Here, we assume to have Complete Spatial Randomness (CSR), i. e. the probability $p_{\text{out\_vote}}$ of any outlier match $m \in \mathcal{O}$, casting a vote on the voting cell v is the same for any voting cell $v \in \mathcal{V}$. Accordingly, the number of outliers voting for a voting cell is binomially distributed. For the GTBL Method, we compare the actual outlier distributions with the ones predicted based on the CSR assumption. Figure 6.1 presents the results. We find our predicted outlier distribution to be sufficiently accurate. While we systematically underestimate the number of voting cells that receive only very few outlier votes, the predicted numbers of voting cells with larger amounts of outlier votes is more accurate. In practice, only the voting cells that received most votes influence the localization success rate. Therefore, the CSR assumption seems to be sufficiently accurate to predict success rates.

Please refer to Section A.3 in the appendix, for the complete derivation of our success rate prediction model. Our final equation is the following:

$$\Pr[N_{\mathcal{I}}^{v_p} \geq 2] = 1 - \left(\prod_{v_i \in \mathcal{V}_{\mathcal{I}}} \left[1 - \left(\Pr[(N_{\mathcal{I}}^{v_i} \geq 2) \cap (N_{\mathcal{M}}^{v_i} = N_{\mathcal{M}}^{v_p})] \right) \right] \right). \tag{6.1}$$

It means that the predicted success rate corresponds to the probably that, among the voting cells that received inlier votes $\mathcal{V}_{\mathcal{I}}$, there is at least one voting cell $v_i \in \mathcal{V}_{\mathcal{I}}$, that obtained two or more inlier votes ($N_{\mathcal{I}}^{v_i} \geq 2$) while it also obtained the most votes of any voting cell, i. e. it obtained the number of votes that the voting peak v_p obtained: $N_{\mathcal{M}}^{v_i} = N_{\mathcal{M}}^{v_p}$.



(a) Tiles

(b) Asphalt (fine)

(c) Asphalt (coarse)

(d) Carpet

(e) Concrete

(f) Wood

Figure 6.1: Evaluation showing the actually observed and the predicted outlier vote histograms, i. e. the ratios of voting cells receiving certain amounts of outliers. Here, evaluated for the GTBL Method on the datasets of the Micro-GPS database [Zhang et al., 2019]. This figure is adapted from [Schmid et al., 2021].

6.3.1 Application of the Prediction Model

To use the model, we require some empirical observations:

- $|\mathcal{V}|$, the number of voting cells with at least one vote;
- $|\mathcal{F}_\mathrm{q}|$, the number of extracted query image features;
- $N_\mathcal{M}$, the number of matches;
- $N_\mathcal{O}$, the number of outlier matches;
- $\mathcal{V}_\mathcal{I}$, the set of voting cells that received inlier votes;
- for all $v_i \in \mathcal{V}_\mathcal{I}$, $p^v_{\text{in\_vote}}$, the probability that a query image feature generates

an inlier vote on v_i.

These model parameter values may vary on every localization attempt. Therefore, we estimate expected values $\mathbf{E}(\cdot)$ of these parameters for the prediction.

$|\mathcal{V}|$ scales with the coverage area size, and is not strongly dependent on the texture type. Assuming similar overlap of the recorded reference images for every application area, to estimate $\mathbf{E}(|\mathcal{V}|)$, we take the average value of $|\mathcal{V}|$ per reference image from evaluations on other application areas, and multiply it with the number of reference images of the current application area.

To find appropriate values for the remaining model parameters, we make use of the small collection of test images. The test images consist of a series of consecutively recorded query images, and some additional overlapping reference images. For every query image among the test images, we perform two evaluations in form of localization attempts using the respective localization method with the examined parameter configuration: the first attempt is part of the *inlier evaluation*, where we use all available overlapping images as reference images; and in a second attempt, for the *outlier evaluation*, the same query image is used, but randomly selected non-overlapping images are used as reference images.

We estimate $\mathbf{E}(|\mathcal{F}_\mathrm{q}|)$ as the average number of extracted query image features from both inlier and outlier evaluations.

For Micro-GPS, the expected number of outliers is independent of the application area size. So it is sufficient to estimate $\mathbf{E}(N_\mathcal{O})$ as the average number of proposed matches during outlier evaluation, because here all matches are outliers. However, for the GTBL Method, where the number of matches scales with the application area size, we measure the average number of outliers per voting cell with at least one vote, and then estimate $\mathbf{E}(N_\mathcal{O})$ through multiplication of that value with $\mathbf{E}(|\mathcal{V}|)$.

The inlier evaluation allows us to estimate the number of inliers. We observe that inlier votes are typically not limited to a single voting cell, but they can be found in a local cluster of voting cells. We count how many inliers we find on average on the voting cell with most inliers $\overline{n_1}$, the voting cell with second most inliers $\overline{n_2}$, and so on. This is done for all voting cells that received at least one inlier vote. Accordingly, we can approximate that there will be a voting cell $v_1 \in \mathcal{V}_\mathcal{I}$ with $\mathbf{E}(N_\mathcal{I}^{v_1}) = \overline{n_1}$ and $\mathbf{E}(p_{\text{in\_vote}}^{v_1}) = |\mathcal{F}_\mathrm{q}|/\overline{n_1}$, a voting cell $v_2 \in \mathcal{V}_\mathcal{I}$ with $\mathbf{E}(N_\mathcal{I}^{v_2}) = \overline{n_2}$ and $\mathbf{E}(p_{\text{in\_vote}}^{v_2}) = |\mathcal{F}_\mathrm{q}|/\overline{n_2}$, and so on.

Furthermore, we estimate

$$\mathbf{E}(N_{\mathcal{I}}) = \sum_{v_i \in \mathcal{V}_{\mathcal{I}}} \mathbf{E}(N_{\mathcal{I}}^{v_i}), \tag{6.2}$$

and therefore

$$\mathbf{E}(N_{\mathcal{M}}) = \mathbf{E}(N_{\mathcal{O}}) + \mathbf{E}(N_{\mathcal{I}}). \tag{6.3}$$

6.3.2 What is a Correct Match of Features?

Counting inliers correctly is a key requirement for the use of the proposed prediction model. In the context of this work, we defined inliers as pairs, consisting of query and reference feature $m \in \mathcal{M} = (f_q \in \mathcal{F}_{\mathrm{q}}, f_r \in \mathcal{F}_{\mathcal{R}})$, that can be used for successful pose estimation. To determine whether we are counting inliers correctly, we observe the number of inliers on the voting peak of successful localization attempts. Localization attempts without any inliers on the voting peak should not succeed.

One approach to determine the correctness of a match would be to determine whether its corresponding pose estimate itself is already correct. However, this underestimates the actual inlier count, e. g. for Micro-GPS, on average there are less than 0.01 matches per localization attempt satisfying this condition, while it achieves a success rate of 90%.

Alternatively, we could take the employed pose estimation approach into account. In our case, both examined localization methods do not use the orientation information of keypoint objects for the final pose estimation. Instead, they determine the query image pose using the voting positions of two (or more) matches. We propose three ways of counting inliers: a) the matches with correct corresponding pose estimate, not considering the orientation error; b) the matches that can, if paired with the right corresponding matches that also vote for the same voting cell, create a correct pose estimate; c) the matches that if paired with a fake keypoint object, which is positioned on the ground truth query image position, create correct pose estimates.

We observe similar inlier counts for all three proposed measures with averages of 7 to 8 inliers per localization attempt of Micro-GPS [Zhang et al., 2019], respectively 76 to 79 for the GTBL Method [Schmid et al., 2020a]. Finally, we decide to treat any match as inlier which is considered to be an inlier by at least one of the three measures.

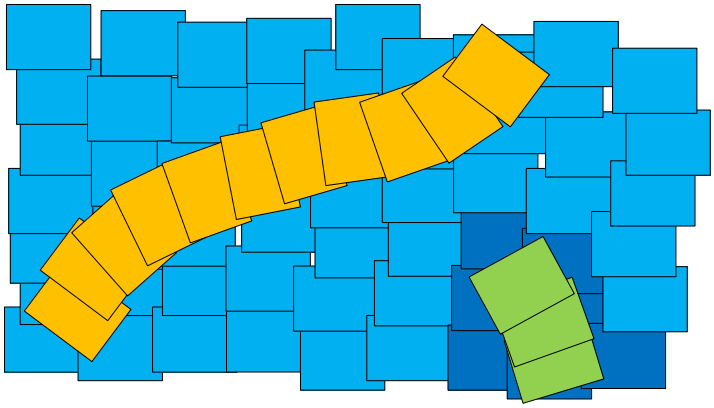


Figure 6.2: Visualization of the types of images that are required to evaluate global localization and local performance. For global localization, a large set of partially overlapping reference images (blue) is used for mapping, and independently recorded query images (orange) are to be localized globally in the map. For parametrization, however, we only evaluate local performance, using a few additional test images (green) and their overlapping reference images (dark blue). This figure is taken from [Schmid et al., 2021].

6.4 Evaluation

In Section 6.4.1, we will first evaluate the suitability of our model to predict the success rate depending on the number of extracted features. Subsequently, in Section 6.4.2, we introduce and evaluate a parameter optimization framework which uses the model to evaluate parameter configuration candidates.

For both evaluations, we use the Micro-GPS database of Zhang et al. [2019] that was recorded with a PointGrey CM3 camera. To determine the actual global localization success rates, for each texture, we use the corresponding 2000 to 4000 partially overlapping recordings as reference images (blue in Figure 6.2), and use 500 separate recordings as query images (orange in Figure 6.2).

In order to predict the value of the global success rate for individual application areas, as described in Section 6.3.1, we require a set of test images. For this, we evaluate localization attempts on 10 additional sequentially recorded query images (green in Figure 6.2), each with a local map of 10 reference images (dark blue in Figure 6.2). For inlier evaluation, we select the 10 closest reference images of the respective query images, and, for outlier evaluation, 10 randomly selected reference images without overlap with the query image.

The set of considered reference images for a query image from the inlier evaluation should include all reference images with significant overlap. Otherwise, we will underestimate the number of inliers. For the Micro-GPS database, taking

the 10 closest reference images is sufficient, as the number of observed inliers does not increase if we use more.

6.4.1 Predicting the Success Rates for Varying Numbers of Extracted Features

Generally, using the procedure described in Section 6.3.1, the model can be used to find suitable values for any parameter. However, two of the most important parameters, directly influencing the computational effort and memory consumption, are the number of extracted features per reference image n_r and per query image $|\mathcal{F}_\mathrm{q}|$. As $|\mathcal{F}_\mathrm{q}|$ is not a free parameter of Micro-GPS, we will focus on n_r.

To find suitable values for n_r, we exploit an advantage of our model-based parameter evaluation strategy: if we are able to estimate the impact of n_r on the model input values, we can predict the success rate for varying n_r values without even having to evaluate them on the test images. Therefore, we evaluate the localization methods on the test images using only a single value of n_r, namely the value suggested by the corresponding authors. Then, we predict the success rate for any n_r value of interest, assuming linear correlation between $N_\mathcal{I}^{v_i}$ (for any $v_i \in \mathcal{V}_\mathcal{I}$) and n_r, and between $N_\mathcal{M}$ and n_r, while assuming constant $|\mathcal{V}|$.

6.4.1.1 Baseline Approaches to Parameter Evaluation

We determine the *global success rate* through exhaustive evaluation of parameter values for the task of localizing the 500 query images on the map. The global success rate is an accurate representation of the actual localization capabilities of a method and provides a good basis for parametrization decisions. Any alternative approach to assessing the localization performance, evaluated on the separate test images, should enable us to make similar judgments about suitable parameter choices, i. e. a similar trend between parameter values and localization performance should emerge.

Besides our prediction model, we evaluate two other approaches to this task.

1. *Local success rate:* based on the previously introduced inlier and outlier evaluation, we propose a simpler model, which only compares voting peaks from the inlier evaluation to that of the This represents the local success rate on the test images. In order to determine the local success rate, we evaluate how often a voting peak from the inlier evaluation receives

at least two inliers and overall more votes than any voting peak observed during the outlier evaluation.

2. *Inlier ratio:* the ratio of inliers among the matches might correlate with the success rate. Therefore, we use it as second alternative approach for performance prediction. Again, this is computed with the inlier and outlier evaluation results.

To predict localization performance with these two alternative approaches, we do not use our predicted model parameters for varying n_r values, but perform inlier and outlier evaluation for every considered value.

6.4.1.2 Results

We evaluate Micro-GPS for n_r values ranging from 5 to 100 with increments of 5, and compute the average prediction errors over these 20 evaluations. For every texture type, we repeat this for 15 different sets of test images, each with 10 query images and their overlapping reference images. Overall, the absolute prediction error of our model for the global success rate is on average 0.217, while it is 0.229 for the local success rate. Figure 6.3 presents texture-specific results, respectively using one test image set. It also presents the inlier ratio. Ideally, the curve of a performance indicator, should be similar to that of the global success rate. However, it is sufficient if it presents similar trends. For example, if the inlier ratio curve would present similar trends as that of the global success rate, it would be suitable to make parametrization choices. But, we observe that, while the general trend of the inlier ratio is often similar to that of the global success rate, its curve is highly volatile, e. g. for fine asphalt in a parameter value range between 5 and 50. One reason for this is the small total number of observed inliers, since we, as mentioned earlier, observe an average of about 7 to 8 inliers per localization attempt of Micro-GPS, and we perform only 10 localization attempts for this evaluation. Additionally, Micro-GPS uses random selection as a feature selection method to reduce the number of considered features per image, which introduces further volatility into the number of observed inliers. As a result, using the inlier ratio as guidance, could lead to suboptimal parametrization or to a situation in which we get stuck in a local maximum. The local success rate curves are more reliable. But, only the curves of our model are smooth, monotonically increasing, and present similar trends as the global success rate. The predicted success rates saturate for larger values of n_r as for the global success rate, which could lead to conservative parametrization choices.

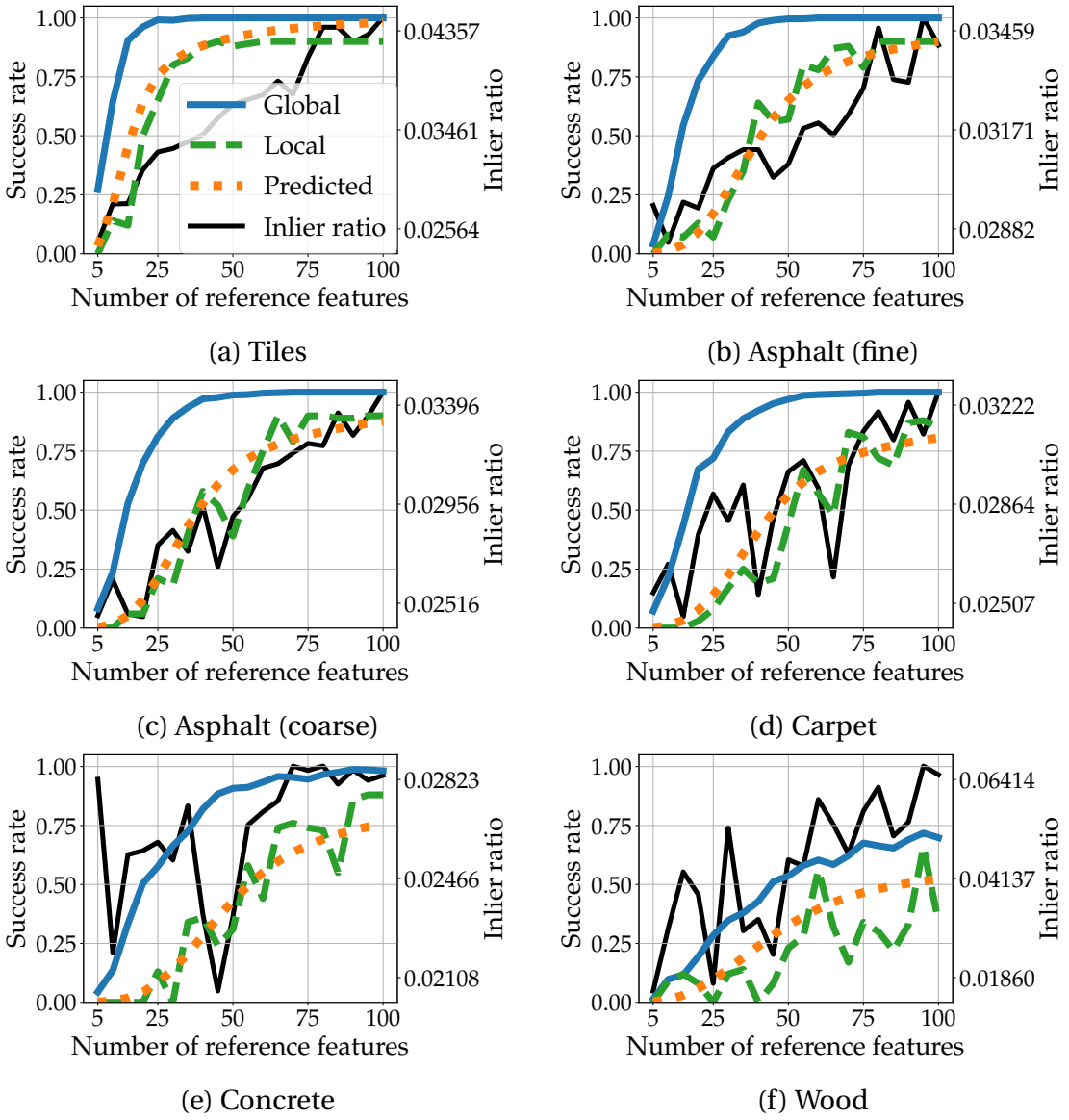


Figure 6.3: Global, local, and predicted success rates, and the inlier ratio, for Micro-GPS [Zhang et al., 2019] using varying numbers of reference features. This figure is adapted from [Schmid et al., 2021].

We evaluate the GTBL Method for n_r values ranging from 50 to 1000 with increments of 50. Again, evaluation is repeated for 15 test image sets. The average error of our model is 0.058, and 0.049 for the local success rate. Figure 6.4 presents some of the results. Our model is accurate for all textures, but wood (Figure 6.4(f)). Closer analysis shows that this is caused by an underestimation of the globally observed number of inliers. So, in this case, the test images were not sufficiently representative for the application area. Apart from wood, the curves of the local success rate are similar to that of the global success rate. However, for concrete, fine asphalt, and carpet, the local success rate overestimates the performance of small n_r values. The inlier ratio curves tend

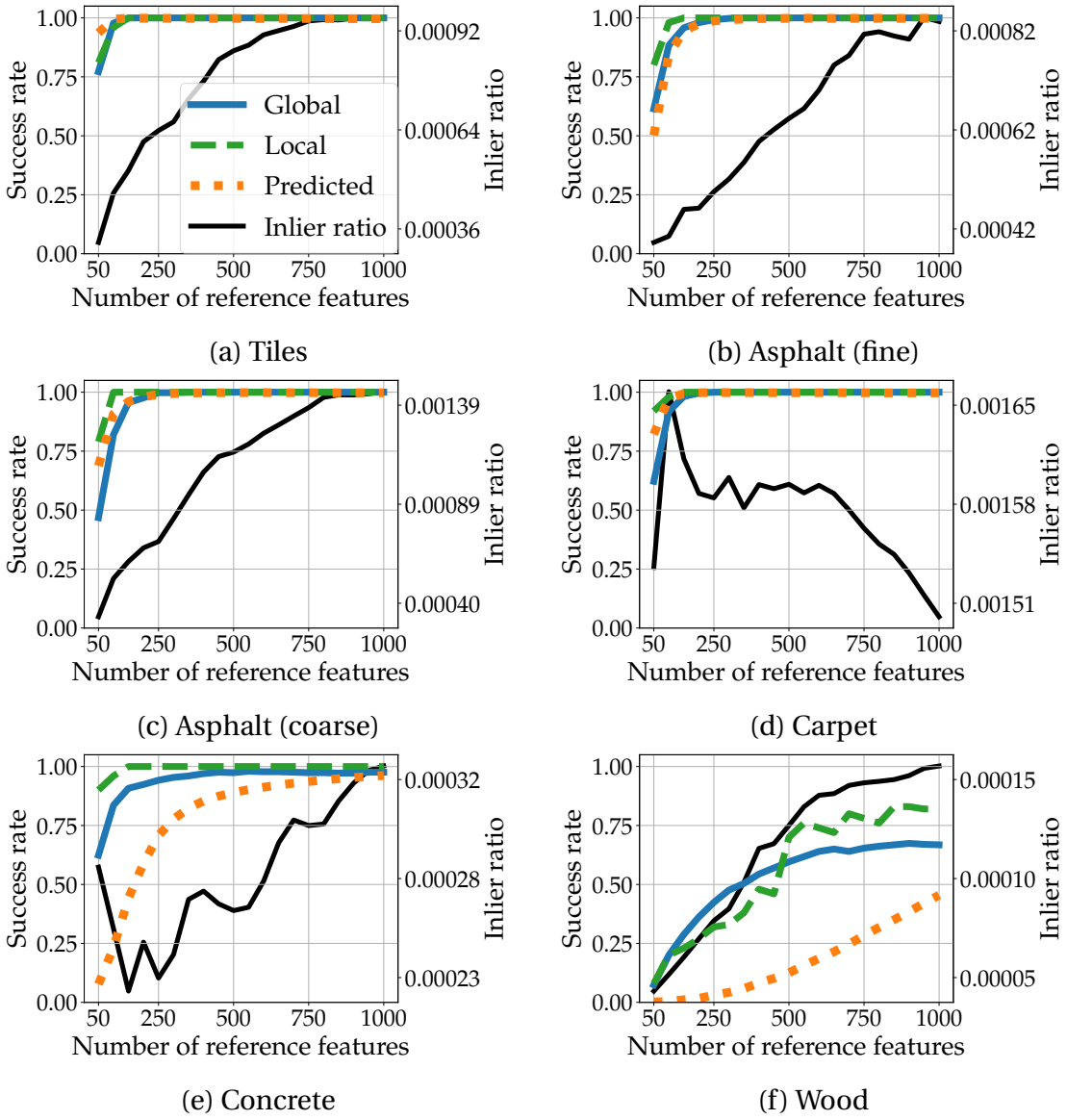


Figure 6.4: Global, local, and predicted success rates, and the inlier ratio, for the GTBL Method [Schmid et al., 2020a] using varying numbers of reference features. This figure is adapted from [Schmid et al., 2021].

to saturate only for significantly larger n_r values as for the global success rate.

Further evaluation shows that having different numbers of query images in the test image sets has no significant effect on our average prediction error. However, the standard deviation of average errors among different sets of test images is affected. Again, for every setting we evaluate 15 test image sets. Using 10 query images, we observe a standard deviation of 0.022 for Micro-GPS and 0.017 for the GTBL Method. For 3 query images, this increases to 0.038, respectively 0.021, and for 15 query images, it decreases to 0.015, respectively 0.013.

So far, to evaluate our model, we estimated expected global numbers of inliers

and outliers based on a single parameter evaluation of n_r, using the default settings the respective authors proposed ($n_r = 50$ for Micro-GPS and $n_r = 850$ for the GTBL Method), and based on the assumption of linear correlation between n_r and the number of inliers and outliers. If, instead, we use the inlier and outlier evaluation for every considered value of n_r, as it is done for the local success rate and the inlier ratio, our average prediction error decreases slightly to 0.199 for Micro-GPS and to 0.048 for the GTBL Method. However, the prediction curves are no longer monotonically increasing.

6.4.2 Using the Model for Parameter Optimization

We propose a simple parameter optimization framework, which uses our success rate prediction model to evaluate possible parameter settings, and apply it to find texture-dependent parameter settings for the GTBL Method [Schmid et al., 2020a].

We select ten important method parameters to be optimized: the number of query image features $|\mathcal{F}_q|$, the number of extracted features per reference image n_r, the histogram cell size of the voting procedure, the number of considered LATCH bits, four parameters of the SIFT keypoint detection method, and two parameters of the LATCH description method. For every parameter, a value range and a step size is defined. This results in a parameter space with more than $5 \cdot 10^9$ possible configurations.

The optimization framework continuously samples random parameter settings from the parameter space, and evaluates them using the prediction model. This means that the localization method is parametrized according to the selected parameter setting, before inlier and outlier evaluations are performed on 10 consecutive query images and their respective reference images to obtain empiric observations, which are then used to estimate the global success rate. Whenever the predicted success rate of a sampled parameter setting is not lower than the predicted success rate of the previously best parameter set minus 0.05, a local optimization is performed for that setting. Here, one of the ten parameters is randomly selected to be optimized, and four different values (two larger and two smaller ones) are tested for it. The four generated parameter settings are then evaluated on the test images, except when the number of extracted reference image features n_r was chosen to be optimized. In that case, we employ the previously described approach of predicting the impact of the parameter change on the model input values and therefore on the resulting success rate. This local optimization procedure is repeated as long

as it increases the predicted method performance, but at least 12 times.

Our optimization framework keeps track of the best-performing parameter setting, considering the predicted success rate and the computation time: a setting is considered superior to another one if its predicted success rate is at least 0.005 higher, or if it is not more than 0.005 smaller but is faster to compute.

6.4.2.1 Results

We run the optimization separately on all six ground textures types. For each texture, the optimization runs for 12 hours on a E3-1270 Intel Xeon CPU at 3.8 GHz. For the final best-performing parameter configurations, the global success rate is evaluated and compared with the results using the default configuration that we suggested in Chapter 5, which we optimized manually in a process lasting several weeks in total. The optimized configurations are very competitive with the default configuration: the mean success rate over all six textures is 94.0% compared to 94.3% with the default, while the mean localization time is reduced from 0.997 s to 0.752 s. We also evaluate 225 randomly sampled configurations per texture. Overall, we observe a mean success rate of 87.1% and a mean localization time of 0.771 s. Particularly for wood, guided parametrization is crucial, as the mean success rate of randomly sampled configurations is only 29.7% compared to 68.6% when using our optimized configuration and 68.2% with the manually optimized default configuration.

Some patterns can be observed in the optimized configurations. On the more difficult textures wood and concrete, the optimized configurations decreased the number of considered LATCH-bits from 15 to 14, while this number is increased to 16 for the other textures. For wood, the most difficult texture, the number of extracted reference and query features are increased from 850 to 1000, respectively from 850 to 950, while on the other textures these numbers are decreased, e. g. to 500, respectively 600, on fine asphalt, reducing the computation time and memory consumption. The difference between our optimized parameter settings and the setting that we previously determined manually seem to be aligned with our analysis that we made in Chapter 5, where we figured that the decreased localization success rate on concrete and wood stems from a lower number of inlier matches. The increased number of extracted features will directly result in an increased number of inliers. Similarly, reducing the number of considered LATCH-bits decreases the number false negatives that were otherwise incorrectly rejected. Anyway, this did not result in an improved success rate of the automatically optimized setting compared to the

manually optimized one, but it represents a more conservative parametrization compared to our manual parametrization for which we chose exactly the right amount of extracted features above which we did not observe significant improvements in the success rate. On the remaining four textures, we already observed success rates of close to 100% and large numbers of correctly matched features. Accordingly, it makes sense that the automatically optimized parameter setting is able to save computation time on these textures, by reducing the number of extracted features, and by increasing the number of considered LATCH-bits, which results in a larger inlier-to-outlier ratio with a smaller overall number of matches to process.

We also applied the local optimization procedure for each texture on the default parameter configuration. This took on average 1550 s, and resulted in comparable optimized parameter settings. The resulting average success rate is 93.9%, while the average localization time is 0.805 s, which, again, is significantly faster than for the default configuration.

On average it took 20.1 s to evaluate a parameter configuration during the examined optimization procedure. Further speedup would be possible through parallelization. In comparison, evaluating the final configuration with all reference images to obtain the global success rate took on average 3347.6 s. Our prediction model thus allows an acceleration by a factor of about 166 in the evaluation of configurations.

6.5 Discussion

We proposed a success rate prediction model for ground texture based localization methods, and used it for a parameter optimization framework. Based on a small collection of test images, our model predicts the global localization success rate, which can be used to optimize parameter settings accordingly.

On the example of the number of extracted features per reference image n_r, we have shown that the predictions are sufficiently accurate for parametrization. Furthermore, due to our model-based approach to the evaluation of parameter configurations, it is not necessary to fully evaluate every considered parameter configuration, because for some parameters such as n_r, we can accurately estimate its impact on the model input values.

Our prediction model can be used to optimize any localization method parameter influencing the localization performance. Accordingly, we were able to build a parameter optimization framework with it, which can quickly evaluate

any considered parameter configuration. Using the configurations obtained from the framework, we achieved a similar localization success rate as with the original default parameter setting, while the localization time was significantly reduced. Here, it was not to be expected that the automatically found parameter settings are better than those that have been determined in a laborious weeks-long manual process, as it was done by us for evaluation of the GTBL Method in Chapter 5. The employed parameter search space is the same in both cases, so both approaches to parametrization are able find good solutions. But, for the manual optimization, we performed the exhaustive evaluation of the global success rate for any considered configuration, which is going to be more accurate than a model-based prediction of the same. Furthermore, a human with domain-knowledge will be able to choose suitable parameter setting candidates, which is likely to be more efficient than the random gradient descent like process we employed for the automatic parameter optimization. Anyway, the advantage of the automated approach should be that a good configuration is found in shorter time and with much less (human-involved) effort. This goal is already achieved by our simple parameter optimization framework, due to the employment of the prediction model.

We assume that the proposed procedure could help with the employment of autonomous agents using the GTBL Method or Micro-GPS for their localization capabilities. For this purpose, the agent may have a specific parameter optimization mode, in which it automatically records a suitable set of the required test images from the destined application area. The agent may then proceed with the optimization itself or it may send the images to a server with more compute power that determines a suitable parameter setting for the specific application area. Of course in a practical scenario, a proper initialization of the parameter search will be more efficient than the random initialization performed by our proposed framework. For example, as we observed in one of our experiments, a simple local optimization starting from a default parameter setting is sufficient to obtain a suitable set of parameters.

7 Deep Metric Learning for Global Localization

Contents of this chapter were partially published in [Radhakrishnan, Schmid, Scholz, and Schmidt-Thieme, 2021] and [Schmid, Simon, Radhakrishnan, Frintrop, and Mester, 2022].

In this chapter, we consider the task of map-based localization, solving Problem 3 (localization). A particularly challenging manifestation of this task is the initial localization, which is necessary when we have no knowledge about the current location of the robot, e. g. after restart or in recovery mode after mislocalization. This task is difficult, because it does not allow to restrict the search space for the current query image pose based on an existing approximate pose estimate. Therefore, features of all available reference images have to be considered in the feature matching step, increasing the computational effort, and increasing the number of incorrectly proposed feature correspondences, which also increases the chance of mislocalization as we have seen in Chapter 5. A possible solution to this problem was introduced by Chen et al. [2018]. With the idea in mind that they would like to consider only those reference images that are actually overlapping with the query image, they propose to apply image retrieval, which is a technique to find similar images to a given query image in a database of reference images [Smeulders et al., 2000]. Then, only the features of the retrieved most-similar reference images are used for the subsequent feature matching and pose estimation steps. Chen et al. propose a Bag of Words (BoW) approach to image retrieval (described in Section 2.4.1.1) that makes use of the hand-crafted SURF [Bay et al., 2006] feature extraction method.

As deep learning approaches have been very successful in computer vision [Russakovsky et al., 2015, Goodfellow et al., 2016]; and in particular, for image retrieval [Arandjelovic et al., 2016, Gordo et al., 2017, Revaud et al., 2019], we propose a deep learning approach to image retrieval of ground images, based on Deep Metric Learning (DML), to substitute the use of hand-crafted feature extractors and the BoW technique. Still, BoW is the current state of the art for the retrieval of ground images, and it was shown to achieve good perfor-

mance (see Chapter 5). This is why we perform an in-depth evaluation of this approach, searching for optimal parametrization to examine the method in its best possible configuration.

The goal of metric learning is to compute descriptors that have small distances between similar data points, e. g. objects of the same class, and large distances between dissimilar data points. While classical metric learning does this by learning a new metric, DML fixes the metric, e. g. Euclidean, but maps the data points to a new feature space, also called embedding space, in which the goal of decreasing the distance between similar data points and increasing it for dissimilar ones is reached [Kaya and Bilge, 2019].

Approaches to DML are often based on Siamese or Triplet networks [Kaya and Bilge, 2019]. During training, these process two, respectively three, input samples simultaneously using identical networks with shared weights. The Siamese network is shown pairs of input samples [Bromley et al., 1994, Chopra et al., 2005], which it aims to map to embeddings with a distance according to the objective function. The Triplet network, on the other hand, is shown three samples: an anchor input sample together with a positive sample, e. g. from the same class as the anchor, and a negative sample [Hoffer and Ailon, 2015].

Our DML method is summarized in Figure 7.1. It learns similarities between ground images to represent them as compact embeddings, i. e. image descriptors, for image retrieval. It consists of a Convolutional Neural Network (CNN) that is trained in Siamese fashion, using an objective function that is adopted from Sánchez-Belenguer et al. [2020]. Subsequently, we employ a k-d tree to find the reference images with most similar embeddings to that of the query image. Our results show that our method outperforms BoW image retrieval, with significantly higher recall values especially for the most difficult cases. Also, we evaluate the use of image retrieval for ground texture based global localization, estimating the query image pose based on the retrieved reference images with most similar image descriptors. On the Micro-GPS database, our GTBL Method (see Chapter 5) performs much better when using image retrievals of our DML method, instead of the retrievals of the BoW approach, and it performs slightly better as without the employment of image retrieval, where all reference images are considered in a brute-force fashion. On the HD Ground Database, we observe significantly increased global localization performance of both StreetMap [Chen et al., 2018] and the GTBL Method, compared to their performance with BoW image retrieval or without image retrieval.

This work contributes a novel deep metric learning approach to represent ground images with compact image descriptors, which are suited for image re-

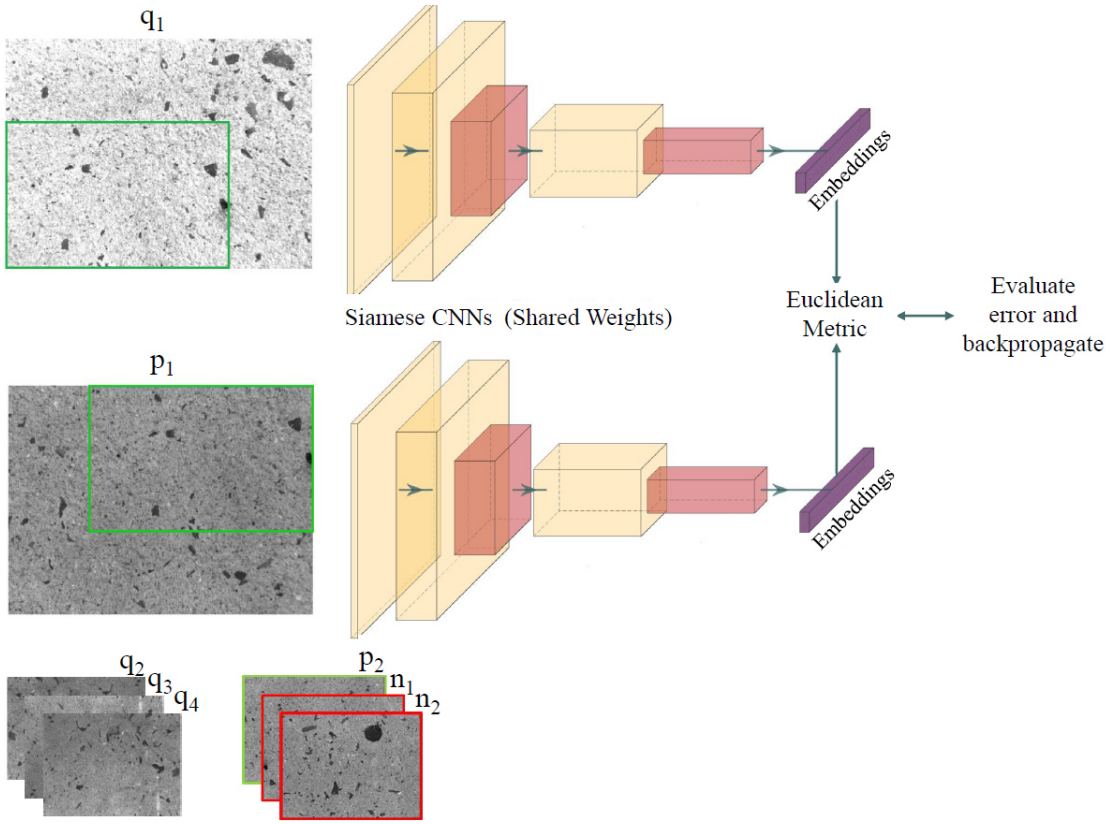


Figure 7.1: Overview of the training procedure of our proposed deep metric learning approach. We use a Siamese Convolutional Neural Network (CNN) architecture, whose final layer activations represent image embeddings. During training, the model tries to predict the overlap between randomly sampled pairs of images. This figure is adapted from [Radhakrishnan et al., 2021].

trieval. We introduce a framework for the evaluation of ground image retrieval, and, we optimize the employed BoW method, the current state-of-the-art approach to ground image retrieval, for the best performing pairing of keypoint detection and feature description method and for the optimal number of considered features per image. Still, we show that our method outperforms BoW both in image retrieval recall and the resulting localization performance.

7.1 Related Work

To the best of our knowledge, we propose the first deep learning approach that can be applied directly to ground image retrieval for ground texture based localization. Existing deep learning approaches to image retrieval are not directly applicable to this task, and would therefore require to be adapted to this domain, which is out of the scope of our work. This is, for example, be-

cause these methods explicitly learn camera poses, like PoseNet [Kendall and Cipolla, 2017], which is not our desired behavior, as we would like to have a solution that generalizes to application areas that have not been seen during training. Furthermore, state-of-the-art methods for image retrieval such as [Revaud et al., 2019], [Gordo et al., 2017], [Noh et al., 2017a], and [Tolias et al., 2016], are trained with a classification loss, having places correspond to classes. Such a classification of images is not trivially applicable to ground texture based localization, because every query image potentially has a different set of overlapping reference images, which means that labels would have to be query image specific.

Zhang and Rusinkiewicz [2018] applied deep learning to ground images, but with the aim of keypoint object detection, which is not in the scope of our work.

The current state of the art for the task of ground image retrieval is the BoW image retrieval method developed by Galvez-López and Tardos [2012]. BoW is an aggregated descriptor [Duan et al., 2015]. The first step for BoW is to build a visual vocabulary. This means that local visual features are extracted from a set of training images, and subsequently clustered into groups of similar features. Each cluster represents a visual word. Then, in order to compute a BoW representation of an image, the visual vocabulary is used to map local visual features from an image to their corresponding visual words, based on their feature descriptors. This mapping allows to quantize continuous feature descriptors and the resulting visual words have lower dimensionality than the feature descriptors. Finally, the query image is represented by the histogram of its visual words, and similar reference images can be found as the ones with most similar histograms.

Besides BoW, there are other aggregation schemes like the Fisher Vector (FV) [Perronnin and Dance, 2007] and the Vector of Locally Aggregated Descriptors (VLAD) [Jégou et al., 2010]. Both FV and VLAD extend the BoW method using information about the statistical distribution of local features. Also, in comparison to BoW, FV and VLAD are able to reduce the dimensionality of the aggregated descriptors [Duan et al., 2015]. However, these methods have not yet been applied to the task of ground image retrieval. They, are not in the scope of this work.

Further related work uses image retrieval for place recognition and other localization tasks. Several methods, such as StreetMap [Chen et al., 2018] and RelocNet [Balntas et al., 2018], follow a coarse-to-fine approach for localization. They retrieve reference images that are similar to the query image globally, followed by a fine-grained adjustment of the estimated query pose, e. g. using

the matches of local visual features.

Gordo et al. [2017] developed a deep metric learning approach for image retrieval, consisting of a CNN trained in Siamese fashion with triplet ranking loss. The authors employ a R-MAC pooling layer that corresponds to a differentiable variant of the R-MAC [Tolias et al., 2016] description method. The triplets generated for training consist of an anchor query image, a positive sample of an image from the same class as the query image, and a negative sample of an image from another class. The network learns to generate query image descriptors that are more similar to that of the positive sample than to that of the negative sample. In the following, we refer to this method as TL-MAC, which stands for triplet loss with R-MAC descriptor.

Revaud et al. [2019] adapt TL-MAC. Instead of using a triplet ranking loss, they directly optimize for the mean Average Precision (AP), considering large numbers of images at each training step. Also, they substitute the R-MAC pooling layer of TL-MAC [Gordo et al., 2017] with a Generalized-Mean (GeM) pooling layer. Accordingly, we refer to this method as AP-GeM.

Sánchez-Belenguer et al. [2020] developed RISE, an image retrieval based indoor place recognizer, which we are building upon with our DML approach. They create a 3D map of the environment with a laser and calibrated spherical camera mounted on a backpack. For image retrieval, they train a CNN in Siamese fashion using overlap information of image pairs. The map is voxelized, and can be used to compute the content overlap of any two images: for each image corresponding depth information is available; therefore, the set of visible mapped-voxels can be identified, and the overlap is then computed as the number of common visible voxels. This does not include voxels through which the visual rays pass, but only those where visual rays end. During training, the network learns to predict image pair overlaps. The activation of the final layers represent the image embeddings. Then, the network is optimized to minimize the error between the predicted dissimilitude (L2-Norm) of the image pairs and their actual common overlap. After training, the network is used offline to compute embeddings for all reference images. Subsequently, to retrieve overlapping reference images during online place recognition efficiently, a k-d tree is used to find the most similar reference image embeddings to the query image embedding.

7.2 The Deep Metric Learning Method

We propose a deep learning framework for the retrieval of overlapping ground images. Our goal is to solve the following problem:

Problem 4 (Ground Image Retrieval) *Given a set of reference ground images $\mathcal{R}$ and a query ground image* q*, retrieve a set of similar reference images to* q *$\hat{\mathcal{R}}_o \subset \mathcal{R}$, including as many images as possible from the set of images $\mathcal{R}_o \subset \mathcal{R}$ that have overlapping content with* q*, i. e. we desire to achieve $\mathcal{R}_o \subseteq \hat{\mathcal{R}}_o \subset \mathcal{R}$.*

7.2.1 Objective Function

Given two images q and r $\in \mathcal{R}$, we normalize them and compute their embeddings e_q and e_r. The distance between q and r is computed with the L2-norm: $d(\mathrm{q},\mathrm{r}) = \|e_\mathrm{q} - e_\mathrm{r}\|_2$, which in our tests worked better than cosine similarity $(e_\mathrm{q} \cdot e_\mathrm{q})/(\|e_\mathrm{q}\|_2 \cdot \|e_\mathrm{r}\|_2)$, which however would have had the advantage of naturally mapping to values between 0 and 1, i. e. the minimal and maximum possible image overlap values.

The actual overlap between the images is represented as $o(\mathrm{q},\mathrm{r})$, it is computed as the physical space that is covered by both images divided by the physical space covered by a single image (we assume all our images to cover the same amount of physical space). The goal of our training procedure is to adapt the weights of our CNN in such a way that $d(\mathrm{q},\mathrm{r}) = 1 - o(\mathrm{q},\mathrm{r})$, e. g. we want to have $d(\mathrm{q},\mathrm{r}) = 1.0$ in the case of no overlap between q and r, and $d(\mathrm{q},\mathrm{r}) = 0.0$ in the case of full overlap. For this purpose, we adopt the overlap loss of Sánchez-Belenguer et al. [2020] as objective function:

$$\mathrm{L} = [d(\mathrm{q},\mathrm{r}) - (1 - o(\mathrm{q},\mathrm{r}))]^2. \tag{7.1}$$

But, in contrast to Sánchez-Belenguer et al., we employ 2D image overlaps, which are available as the ground truth poses and the sizes of the areas covered by the images are known for training images.

7.2.2 The Model and its Application

Our network architecture and its training procedure is illustrated in Figure 7.1. It consists of a CNN that extracts image features. The activations of the final fully-connected layer represent the image embeddings. We examine two variants

of our architecture, one using ResNet-50 [He et al., 2016] as CNN backbone, and the other using DenseNet-161 [Huang et al., 2017]. The proposed ground texture DML models, using ResNet and DenseNet backbones are henceforth referred to as *DML-R* and *DML-D* respectively.

During training, input image pairs are processed in Siamese configuration. The network is trained with positive samples of actually overlapping pairs, (q_1 with p_1 and q_2 with p_2 in Figure 7.1), and negative samples of non-overlapping pairs, (q_3 with n_1 and q_4 with n_2 in Figure 7.1). For each training sample, the loss is computed according to Equation (7.1) and backpropagated.

For the image retrieval system, a k-d tree is built from all reference image embeddings. Then, at inference time, it can be used to retrieve the reference images with most similar embeddings to that of the query image.

7.2.3 Implementation

We implement our deep metric image retrieval approach in PyTorch [Paszke et al., 2019], based on the Siamese network approach for image similarity with deep ranking of Wang et al. [2014]. Our ResNet-50 [He et al., 2016] and DenseNet-161 [Huang et al., 2017] backbones are pre-trained on ImageNet [Russakovsky et al., 2015] for the task of object classification. Subsequently, they are trained in Siamese configuration for ground image retrieval on the Micro-GPS database [Zhang et al., 2019], respectively on the HD Ground Database [Schmid et al., 2022]. For more compact image embeddings, we examined the replacement of the CNN's final pooling layers with generalized mean pooling layers [Radenović et al., 2019, Cao et al., 2020]. However, in our tests on the Micro-GPS database this decreased performance. So, we maintain the adaptive average pooling layers. Also, we experiment with node sizes of the final layer, which defines the embedding size, of 1000, 2048, and 4096. We observe best performance with an embedding size of 1000.

During training, we employ a batch size of 64 and tune the network weights using Adam optimizer with a learning rate of 10^{-4}, a weight decay of 10^{-5}. After training, for image retrieval, we find the k reference images with most similar embeddings to the query image, using the scikit-learn [Pedregosa et al., 2011] k-d tree.

7.3 Evaluation

Our main evaluation of the proposed image retrieval approach is done in Section 7.3.2 on the Micro-GPS database [Zhang et al., 2019]. Here, we make an in-depth examination of the image retrieval performance, using the performance metrics described in Section 7.3.1, and we examine the performance in the case of an application for ground texture based localization. In Section 7.3.3, we extend this evaluation with an examination of the proposed method being applied for localization on the HD Ground Database [Schmid et al., 2022].

7.3.1 Performance Metrics

We use recall as image retrieval performance metric. Recall describes the share of correctly retrieved images. It depends on the overall number of retrieved images $k = |\hat{\mathcal{R}}_o|$, i. e. the k reference images with most similar descriptors to that of the query image, and the number of actually available correct retrievals, i. e. the number of reference images with overlap with the query image, which can be a different number for each query image. So, as a function of k, we define the recall as:

$$\mathrm{R@}k = \frac{\#\,\text{correctly retrieved images}}{\#\text{number of actually available correct retrievals}}. \tag{7.2}$$

Whether a retrieved reference image is considered to be correct depends on its overlap with the query image. Generally, we are interested in all reference images with any overlap, but the ones with large overlap are the most valuable ones for the localization task, as they contain potentially the most correspondences of local image features with the query image. This is why we compute R@k for varying overlap thresholds. $\mathrm{R}_x\mathrm{@}k$ represents the share of correctly retrieved reference images that have at least $x\,\%$ overlap with the query image.

Finally, we employ the image retrievals for ground texture based initial localization. Here, we evaluate the localization success rate as proposed by Zhang et al. [2019], where localization attempts are considered to be correct if the estimated query image pose has a translation difference of less than 4.8 mm and an absolute orientation difference of less than 1.5 °.

7.3.2 Evaluation on the Micro-GPS Database

This section presents our evaluation on the PointGrey CM3 Micro-GPS ground image database [Zhang et al., 2019] that was published in [Radhakrishnan et al., 2021].

In addition to our DML methods, we evaluate BoW as the current state-of-the-art approach, and we consider the deep metric metric learning approaches TL-MAC and AP-GeM, which have been introduced in Section 7.1. These methods are trained with a classification loss, which, as explained earlier, prevents us from being able to train them directly for the task of image retrieval for ground texture based localization. However, they have been found to have good generalization capabilities [Pion et al., 2020], as they outperform other methods for visual localization tasks without being trained on the evaluation database. Accordingly, we employ these methods using pre-trained weights[1]. For TL-MAC the model was trained on the Landmarks-clean dataset [Babenko et al., 2014]. For AP-GeM, we achieve the best recall using weights of an instance that was trained on the Google-Landmarks Dataset [Noh et al., 2017b], which contains more than one million images from 15000 places. Furthermore, we examine two baseline approaches. The first is *Random*, sampling k reference images as retrieval result. Our second baseline is to use the ResNet-50 [He et al., 2016] and DenseNet-161 [Huang et al., 2017] CNNs without task-specific fine tuning, i. e. they are only pre-trained on ImageNet [Russakovsky et al., 2015].

In Section 7.3.2.1, we describe the preparation of data for the training procedure of the DML networks. Implementation details of the BoW method are presented in Section 7.3.2.2, this includes a survey about the optimal choice of the detector-descriptor pairing. Subsequently, we examine image retrieval performance in Section 7.3.2.3 and performance on the task of initial localization in Section 7.3.2.4. In all cases, the number of retrieved most similar reference images is fixed to $k = 100$.

The BoW approach is evaluated on a Intel Core i5-3570 CPU with 32GB RAM and 6 cores at 3.40 GHz, while our CNNs are trained and evaluated on five Titan X Pascal 12GB GPUs and an Intel Xeon E5-2630 v4 CPU at 2.20 GHz.

7.3.2.1 Data Preparation for Training

We separate a sequence of 500 query images per texture for the evaluation, and use the other for parameter optimization and network training.

[1] https://github.com/naver/deep-image-retrieval

In order to prepare the data for training of our proposed DML-D and DML-R models, we compute the pairwise overlap of each query image with all reference images. This allows to identify the positive training samples of query-reference image pairs for which we require to have at least 20% overlap, because we observed that the models can get confused by low-overlapping positive samples, which seem to be hard to distinguish from negative samples without any overlap. We train our models jointly on all textures, because we aim for generalized models. However, the number of available positive samples varies for the different textures: roughly 3000 for concrete, tiles, and wood, and roughly 10 000 for carpet, coarse, and asphalt. To create additional training samples, we apply random image augmentations of flips and rotations between ± 45 degrees. Finally, we obtain about 185 000 positive samples, and the same number of negative non-overlapping samples is prepared. These pairs are shuffled to be processed in random order to avoid processing multiple similar inputs in a row.

7.3.2.2 Implementation of the BoW Approach

We implement the BoW approach, using the FBOW library[2] to create the vocabulary, using the library's default configuration, and we employ the OpenCV 4.0 [Bradski, 2000] library to extract the required local visual features. Here, an important hyper-parameter choice is the type of the employed local visual features. We examine the keypoint detectors and feature description method that we found to be the most successful ones for ground images in Chapter 4: SIFT [Lowe, 2004], SURF [Bay et al., 2006], and AKAZE [Alcantarilla et al., 2013] are employed both as methods for detection and description, and additionally we examine the binary description methods LATCH [Levi and Hassner, 2016] and BRIEF [Calonder et al., 2010]. The methods are parametrized with the corresponding optimized parameter settings of Chapter 4 that can be found in the appendix Section A.1. For the vocabulary creation, a large set of features is required from the application domain. For this purpose, we choose to extract 1000 features per image from 1000 reference images per texture. In order to limit the number of extracted features per image, we employ NMS, i. e. we choose the features with largest keypoint response values that have been assigned by the respective keypoint detectors. Subsequently, the vocabulary is used to map images to BoW representations using their respective set of extracted features. Here, another important hyper-parameter choice is the size of this feature set n. Again, we implement this choice by taking the features with largest keypoint response values. In the following, we examine the optimal choice of the

[2] https://github.com/rmsalinas/fbow

Table 7.1: R@100, i. e. the recall when retrieving 100 reference images per query image, evaluated for the Bag of Words (BoW) approach on the carpet texture of the Micro-GPS database [Zhang et al., 2019] for varying detector-descriptor pairings.

Detector	Descriptor	R@100(%)
SIFT	SIFT	**35.9**
SURF	SURF	9.4
AKAZE	AKAZE	17.4
SIFT	LATCH	14.1
AKAZE	BRIEF	12.1
AKAZE	LATCH	13.7

detector-descriptor pairing.

BoW Parameter Optimization: First, we set $n = 1000$ and vary the detector-descriptor pairings. We evaluate on the carpet texture and present results of R@100 in Table 7.1. The combination of SIFT keypoint object detection and SIFT feature description clearly outperforms the other options. Hence, we use this variant in the following.

We also investigate the texture-specific optimal choice of n, for values between 100 and 1000 with a step size of 100. The results can be found in the Appendix (Section A.4). To achieve optimal BoW retrieval performance, we will always evaluate BoW image retrieval by selecting the respective texture-specific optimal choice of n.

7.3.2.3 Image Retrieval Performance

Figure 7.2 presents the results for R_0@100. The random baseline has the lowest recall results. Varying performance of this method for the different textures can be explained by the correspondingly varying number of reference images. Our models, DML-D and DML-R, have the best retrieval performance, clearly outperforming BoW, the current state-of-the-art for ground image retrieval. DML-D, with the DenseNet-161 backbone, achieves slightly better performance than DML-R, using a ResNet-50 backbone. Also, we observe that fine-tuning the models for the use on ground images is of great importance, as our DML methods perform much better than the networks that have not been trained on ground images: ResNet, DenseNet, TL-MAC, and AP-GeM. This can be explained by the fact that the random patterns observed in ground images are quite different to the structured environments of, for example, ImageNet.

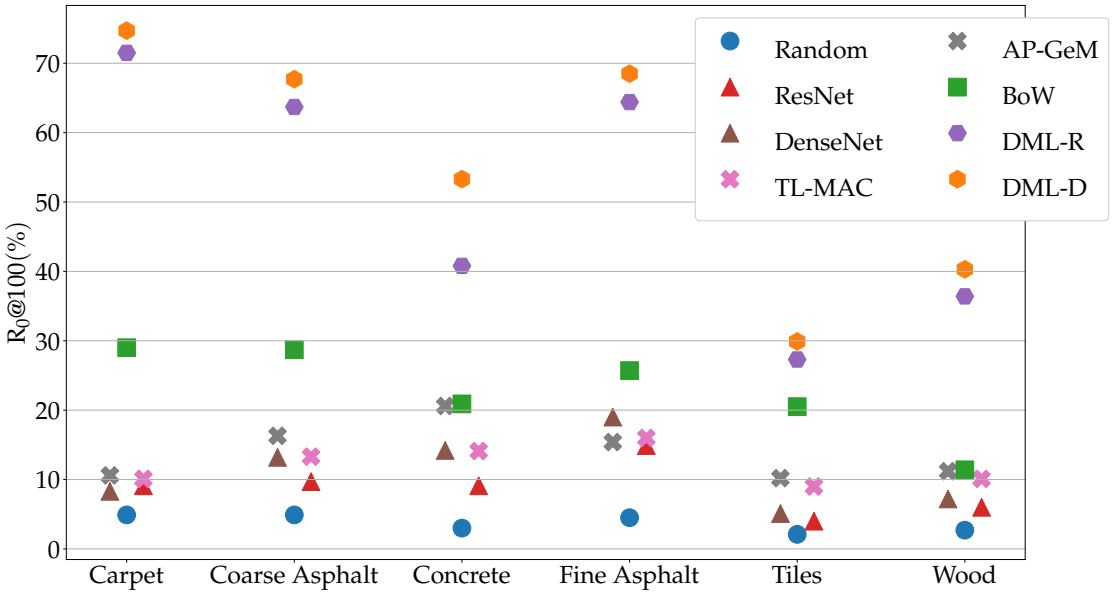


Figure 7.2: Texture-specific R@100 performance, i. e. the recall when retrieving 100 reference images per query image, for all textures of the Micro-GPS database [Zhang et al., 2019], evaluated for our proposed methods DML-R and DML-D, the baseline approaches random and Bag of Words (BoW), and the deep neural networks that have not been trained specifically for our task ResNet-50 [He et al., 2016], DenseNet-161 [Huang et al., 2017], TL-MAC [Gordo et al., 2017], and AP-GeM [Revaud et al., 2019]. This figure is adapted from [Radhakrishnan et al., 2021].

Matching the results of Zhang et al. [2019] and Schmid et al. [2020a] (presented in Chapter 5), we identify concrete and wood to be the most challenging of the evaluated textures, but Zhang et al. [2019] and Schmid et al. [2020a] observed good results on tiles. However, concrete, wood, and tiles are also the textures for which we have only about 3000 samples of overlapping query-reference image pairs (without synthetic augmentation), while we have 10 000 for the others. This might be the main reason for our poor performance on tiles and it adds to the challenge on concrete and wood.

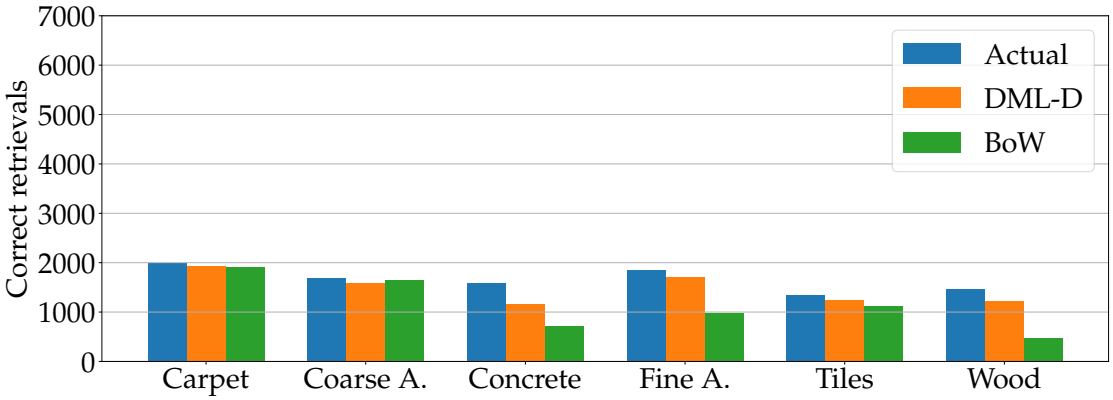
Table 7.2 presents R@100 averaged over all textures. Different thresholds for the minimum required overlap of the retrievals are considered, i. e. $[R_0,..,R_{80}]$@100, where R_x@100 is the value of R@100 considering only those retrievals with greater (not equal) than x% overlap as correct. Our DML-D model has the best retrieval performance with an average R_x@100 of 83.3% (averaged over $x \in \{0, 20, 40, 60, 80\}$), outperforming the BoW approach with an average of 61.2%. The reference images with largest overlap are mostly retrieved correctly by the BoW approach; hence, it achieves a R_{80}@100 of 93.5%. However, most of the reference images with only small amounts of overlap with the query

Table 7.2: For varying values of x, R_x@100 results averaged over all textures of the Micro-GPS database [Zhang et al., 2019], i. e. the recall performance when retrieving 100 reference images per query image and when considering only those retrievals with greater than x% overlap as correct. The performance is evaluated for our proposed methods DML-R and DML-D, the baseline approaches random and Bag of Words (BoW), and the deep neural networks that have not been trained specifically for our task ResNet-50 [He et al., 2016], DenseNet-161 [Huang et al., 2017], TL-MAC [Gordo et al., 2017], and AP-GeM [Revaud et al., 2019]. The respective best recall results are highlighted in **bold**.

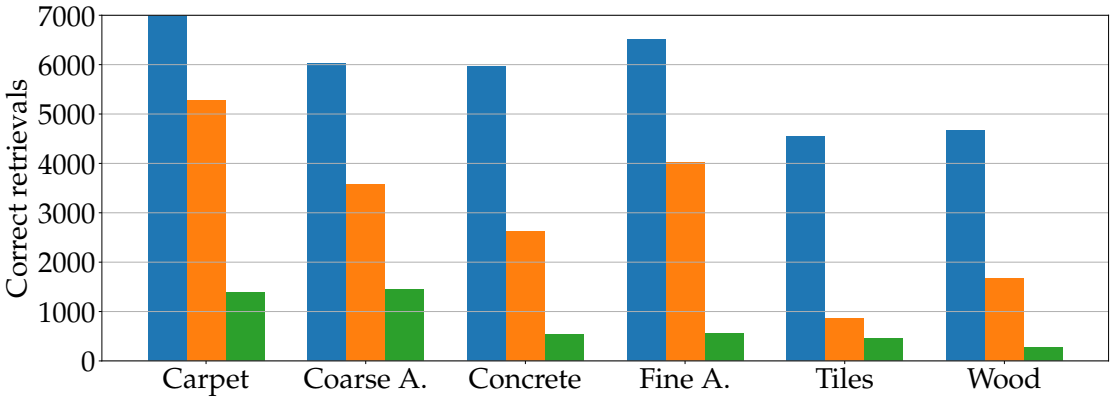
Model	R_x@100(%)				
	R_0	R_{20}	R_{40}	R_{60}	R_{80}
Random	3.7	3.7	3.7	3.8	4.0
ResNet	8.8	11.3	14.0	17.1	25.2
DenseNet	11.2	15.9	20.8	25.5	41.5
TL-MAC	12.2	17.4	22.4	28.7	41.3
AP-GeM	14.1	18.9	23.6	29.0	39.0
BoW	22.7	42.3	64.4	82.9	93.5
DML-R	51.0	68.0	81.9	90.5	93.5
DML-D	**55.7**	**75.0**	**89.5**	**97.0**	**99.3**

image are not correctly retrieved, which leads to poor recall values for R_0@100 and R_{20}@100 of only 22.7%, respectively 42.3%. We investigate this further by comparing the numbers of correctly retrieved reference images with at least 40% overlap (Figure 7.3a), and with less than 40% overlap with the query image (Figure 7.3b). Figure 7.4 presents examples where DML-D correctly retrieved images with more, respectively less, than 40% overlap. As expected, we observe BoW to be competitive with DML-D for retrieving the reference images with large overlaps, while it gets outperformed by DML-D for reference image with small overlaps. This indicates a better representation of our learned image embeddings compared to the BoW image representations. Summarizing the results of all six textures, a maximum number of 34944 reference images with less than 40% overlap could have been retrieved. DML-D retrieved 51.6% (18040) of them and BoW 13.4% (4678). In contrast, of the 9921 possible retrievals with at least 40% overlap, DML-D retrieved 89.4% (8874) and BoW 69.3% (6878).

It is also of interest to examine how often image retrieval failed completely, i. e. not a single overlapping reference image is retrieved, because in these cases subsequent successful pose estimation based on the retrieved images is impossible with any localization method. For DML-D, we observe a total of 18 failure cases, 17 on concrete and one on wood. The BoW approach has 201 failure cases, 135 on wood, 53 on concrete, 7 on tiles, 5 on fine asphalt, and one on coarse asphalt.



(a) Considering only the reference images with $\geq 40\%$ overlap as correct.



(b) Considering only the reference images with $< 40\%$ overlap as correct.

Figure 7.3: Evaluation on the datasets of the Micro-GPS database [Zhang et al., 2019] (carpet, coarse asphalt, concrete, fine asphalt, tiles, and wood): the number of actually available correct reference image retrievals that have $\geq 40\%$ (a), respectively $< 40\%$ (b), overlap with the tested query images, and the number of those reference images that is retrieved by our DML-D method and the Bag of Words (BoW) method. These figures are adapted from [Radhakrishnan et al., 2021].

7.3.2.4 Initial Localization Performance

Finally, we evaluate the localization success rate of initial localization, using BoW and DML-D image retrievals. Here, we employ the GTBL Method and StreetMap [Chen et al., 2018], considering only the retrieved reference images for potential feature correspondences with the query image. We examine the localization success rate in four modes:

1. using all available references images, i. e. without (w/o) image retrieval;
2. using the ground truth (GT) set of actually overlapping reference images;
3. using the top-100 image retrievals of the BoW approach;
4. using the top-100 image retrievals of our DML-D model.

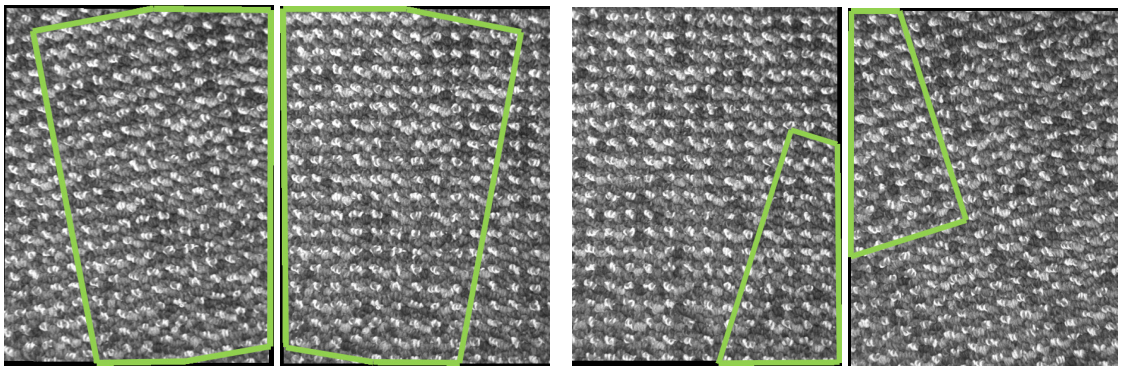


Figure 7.4: Two example image pairings from the carpet texture of the Micro-GPS database [Zhang et al., 2019] of a query image and the reference image that the DML-D method determined to be the most similar one to that query image. The image pairing on the left has an overlap of 75.8% and the one on the right has 19.9% overlap. The green border indicates the overlapping area. This figure is taken from [Radhakrishnan et al., 2021].

Also, for StreetMap, we include the results that we obtained in Section 5.2.1, where the BoW approach was applied with a different setting. In this case, we used SURF features [Bay et al., 2006], instead of SIFT features [Lowe, 2004], as we attempted to re-implement the technique as it was applied in the original StreetMap implementation [Chen et al., 2018]. In the following evaluation, we call this particular variant BoW-SURF. For BoW-SURF, instead of retrieving only 100 reference images, we considered the 20% of reference images with most similar BoW representations, which is a number ranging from 403 for carpet to 809 for tiles. Another difference is the number of extracted features n used to create the BoW representations, here we always used $n = 1000$ features instead of selecting a texture-specific value as it is done for our optimized settings that we obtained in this chapter. Nevertheless, the used SURF features of BoW-SURF were optimized on the Micro-GPS database [Zhang et al., 2019] to be used for image retrieval and subsequent feature-based localization with StreetMap (see Section 5.2.1).

The initial localization success rates of the evaluated methods are presented in Table 7.3. Across all textures and for both evaluated methods, we observe improved localization success rates when using our DML-D retrievals instead of BoW retrievals. This is an improvement from 87.3% to 96.6% for the GTBL Method, and from 90.0% to 97.1% for StreetMap, when using our optimized BoW settings. For comparison, the average success rate with AP-GeM is 49.9%.

The performance of StreetMap with our texture-specifically optimized settings is still better as with BoW-SURF. This is even though we observe it to be bene-

Table 7.3: Initial localization results of the GTBL Method [Schmid et al., 2020a] and StreetMap [Chen et al., 2018] on the Micro-GPS database [Zhang et al., 2019], using different methods to determine the set of reference images that is considered for localization of a given query image. We evaluate the performance without (w/o) image retrieval, when using the ground truth (GT) set of actually overlapping reference images, and when using the retrievals of DML-D and Bag of Words (BoW). For StreetMap, we also present the results when using the BoW not as described in this chapter, but as in Section 5.2.1 using SURF features (BoW-SURF).

Texture	Method	Success rate (%)				
		w/o	GT	DML-D	BoW	BoW-SURF
Carpet	GTBL Method	100	100	100	99.8	
	StreetMap	100	100	100	100	95.0
Coarse Asphalt	GTBL Method	100	100	100	99.8	
	StreetMap	100	100	100	99.8	98.0
Concrete	GTBL Method	97.8	100	92.8	81.8	
	StreetMap	100	100	96.6	88.4	82.0
Fine Asphalt	GTBL Method	100	100	100	98.2	
	StreetMap	100	100	100	99.0	98.0
Tiles	GTBL Method	100	100	99.8	98.4	
	StreetMap	100	100	99.8	98.6	99.2
Wood	GTBL Method	75.0	97.4	87.0	45.6	
	StreetMap	74.8	97.4	86.4	54.2	39.0
Average	GTBL Method	95.5	99.6	96.6	87.3	
	StreetMap	95.8	99.6	97.1	90.0	85.2

ficial to the success rate of StreetMap to retrieve larger numbers of reference images, which should be in favor of BoW-SURF, which retrieves a significantly larger number of images. This observation is also in line with StreetMap having better performance without image retrieval as with the employment of BoW retrievals. It seems that a significant number of overlapping reference images are missed by the BoW approach. The same is true for the GTBL Method. However, image retrieval has the potential to move the success rate of both localization methods to almost 100%, as we obtain the largest success rates when considering only the ground truth of actually overlapping reference images. And, as a matter of fact, using our DML-D retrievals improves the overall success rate of both methods compared to the setting without image retrieval. A look at the detailed results shows that the success rates on concrete are actually better without retrieval, but the performance on wood is much better using DML-D retrievals, which leads to an overall improvement of the average success rates. The improved performance on wood might also be valued higher, as it presents itself as the most challenging texture for ground texture based localization.

Furthermore, the application of image retrieval also has the advantage of reduc-

ing the required computation time for localization. In the case of StreetMap, localization without image retrieval takes on average 12.98 s, where more than 99% (12.87 s) of this time is spent on feature matching. With our DML-D retrievals, on the other hand, localization takes on average 0.57 s, of which feature matching still takes more than 80% (0.46 s). Similarly, in the case of the GTBL Method, we observed in Section 5.2.1.2 that using just 100 instead of all 2014 references images of the carpet texture reduces the computation time for feature matching from 286.47 ms to only 15.25 ms. A robotic agent using our image retrieval method could therefore localize faster, which can be highly beneficial in practice.

7.3.3 Evaluation on the HD Ground Database

This section presents some results of [Schmid et al., 2022] for initial localization on the HD Ground Database.

Here, we evaluate StreetMap and the GTBL Method in the following variants:

- **GTBL Method:** All reference images are considered.
- **StreetMap BoW** and **GTBL Method BoW:** Using BoW image retrieval, only the 15 reference images with most similar BoW representations are considered.
- **StreetMap DML** and **GTBL Method DML:** Only the 15 reference images with most similar CNN embeddings of DML-D are considered.

7.3.3.1 Training of the DML-D Model

We take the DML-D network that we already trained jointly on all textures of the Micro-GPS database [Zhang et al., 2019], and fine-tune it on the training areas of the HD Ground Database. Training on the Micro-GPS database was done with a total of 370 000 image pair samples. For the HD Ground Database, again using the image augmentations of flipping and rotating, we obtain about 112 000 positive and negative training samples, i. e. pairs of overlapping, respectively non-overlapping, image pairs. Therefore, we perform training with 224 000 additional samples from the HD Ground Database.

7.3.3.2 BoW Parameters

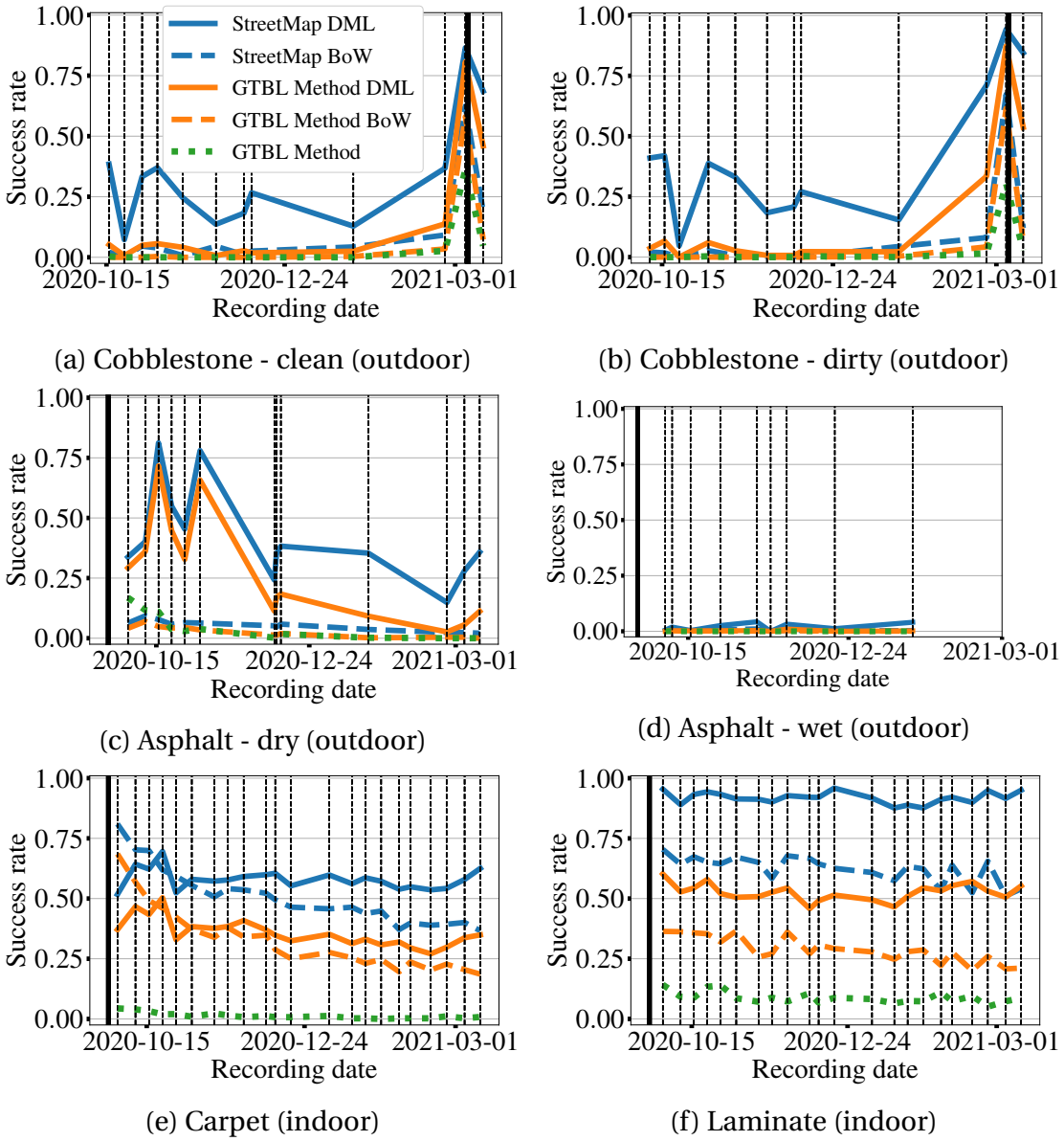
For the BoW approach, we employ SIFT with the same texture-specific parameter settings optimized on the respective training areas that were also employed for image stitching to create the maps of the HD Ground Database. Vocabularies are created texture-specifically, using up to 1000 features per reference image of the training areas.

7.3.3.3 Results

Figure 7.5 presents success rates on the regularly recorded test sequences. The success rate is relatively stable over time for both indoor textures, while it highly depends on the date of recording for the outdoor textures, with higher success rates closer to the date of mapping, which is indicated by the thick vertical line. Using image retrieval to reduce the number of considered reference images improves the success rates. StreetMap and GTBL Method mostly achieve their highest success rates with our DML Image Retrieval approach, while StreetMap reliably outperforms the GTBL Method.

We observe lower success rates on the cleaned cobblestone area, compared to the area being recorded without cleaning, e. g. for the GTBL Method, the mean success rate of its variant without image retrieval changes from 8.1% to 7.7% on the cleaned area and with DML Image Retrieval it changes from 16.6% to 14.1%, and for StreetMap DML it changes from 40.9% to 33.7%. However, these changes are much smaller than the changes in success rate that occur for varying recording dates. Therefore, cleaning might have little influence and the lower performance on the cleaned variant might be explained by fluctuations that are to be expected when examining different parts of an application area. On the other hand, to our surprise, at corresponding recording dates, we observe similar success rates for the cleaned and not-cleaned variants. This means that the larger and smaller success rates, that we observe for varying recording dates, are, in addition to the time interval to the moment of mapping, largely explained by the overall condition the application area was in at the point of recording, e. g. the humidity of the ground. Otherwise, we would expect to observe random fluctuations for individual image sequences recorded at the same date.

On the wet asphalt sequences, average success rates drop to 2% for all examined methods. It seems to be difficult to identify feature correspondences between the wet asphalt test images and the map that was recorded at dry condition.



(a) Cobblestone - clean (outdoor)

(b) Cobblestone - dirty (outdoor)

(c) Asphalt - dry (outdoor)

(d) Asphalt - wet (outdoor)

(e) Carpet (indoor)

(f) Laminate (indoor)

Figure 7.5: Initial localization success rates on the four main textures of the HD Ground Database [Schmid et al., 2022] (cobblestone (clean and dirty), asphalt (dry and wet), carpet, and laminate), for varying time intervals between the recording date of the reference images (indicated by the thick vertical line) and the test sequences (indicated by dashed vertical lines). Note that the reference images for cobblestones were recorded later than most of the test sequences, while the opposite is true for the rest of the textures. The success rates are evaluated for StreetMap [Chen et al., 2018] and the GTBL Method [Schmid et al., 2020a], using DML-D and Bag of Words (BoW) to retrieve the considered reference images, and in the case of the GTBL Method also without image retrieval, i. e. all reference images are considered. This figure is adapted from [Schmid et al., 2022].

This is in line with our previous explanation that the success rates seem to be highly dependent on the condition of the ground at the time of recording.

7.4 Discussion

We introduced a deep learning approach to image retrieval of ground images, using a CNN trained in Siamese fashion for the task of predicting the overlap of image pairs. Our method significantly outperforms BoW [Galvez-López and Tardos, 2012], a state-of-the-art method relying on hand-crafted features, which was proposed by Chen et al. [2018] to be applied for the task of retrieving overlapping ground images. Also, the image retrievals of our method are significantly better suited for initial localization than that of the BoW approach.

In this work, we examined generalized models, being trained jointly on all textures. We also examined the performance of our models if trained texture-specifically, which did not clearly improve the image retrieval recall. The resulting localization success rate on the Micro-GPS database of Zhang et al. [2019] was even slightly lower (96.0% to 96.6%).

Compared to a setting in which all reference images are considered for initial localization, using the image retrievals of our method lead to slightly increased success rates on the Micro-GPS database, and significant improvements on the HD Ground Database [Schmid et al., 2022]. Better success rates with image retrieval can be explained with the smaller number of considered reference images without overlap with the query image that will only contribute incorrect feature matches to the localization procedure. While we observed this advantage on the Micro-GPS database, this effect seems to be even more significant on the challenging HD Ground Database, in which the number of reference images are larger, and in which we recorded test sequences over a long period of time.

Similarly, the difference in localization performance between the GTBL Method [Schmid et al., 2020a] and StreetMap [Chen et al., 2018] are emphasized on the HD Ground Database. On the Micro-GPS database, both methods have more than 95% success rate without retrieval and slightly higher success rates with our DML-D retrievals. Still, StreetMap tends to perform better than the GTBL Method. This difference increases on the more difficult HD Ground Database.

Additionally to the improvements on the localization success rate, the employment of image retrieval for initial localization reduces the computational effort and the resulting required time for localization significantly. For a holistic contemplation, however, we would have to include the required computational effort for image retrieval.

Overall, it seems that the employment of image retrieval for initial ground

texture based localization is a promising approach. While we observed volatile performance with the BoW approach, where the retrieval performance highly depends on used feature extractor, its parametrization, and the number of extracted features, our DML-D approach, being trained jointly on all textures, ensures good results across all textures of both databases and for both evaluated localization methods.

For future research, we would like to investigate the generalization performance to textures not being included in the training process.

8 Faster Local Localization with Keypoint Sampling

Contents of this chapter were partially published in [Schmid, Simon, and Mester, 2020b] and [Schmid, Simon, Radhakrishnan, Frintrop, and Mester, 2022].

In previous chapters, we have seen that state-of-the art feature-based ground texture based localization methods like Micro-GPS [Zhang et al., 2019], StreetMap [Chen et al., 2018], and the GTBL Method [Schmid et al., 2020a] enable reliable centimeter precise positioning on several common ground textures. However, these methods induce a significant computational load.

Zhang et al. identified matching as one of the most time-consuming steps of the task, so they propose to use an Approximate Nearest Neighbor (ANN) search index [Zhang et al., 2019].

In our previous work [Schmid et al., 2020a], presented in Chapter 5, matching is further sped up with the identity matching approach, which matches features only if their descriptors are identical. This also allows to take advantage of prior pose estimates to consider only the closest reference images during feature matching, which cannot be done with ANN matching, because the search index is built globally for all reference images.

Based on the identity matching technique and compact binary descriptors, we proposed the GTBL Method. This method outperforms Micro-GPS [Zhang et al., 2019] if it can take advantage of an available prior pose estimate [Schmid et al., 2020a].

However, one remaining bottleneck, preventing the real-time applicability of ground texture based localization, is keypoint detection. Previously in Chapters 4 and 5, we found the SIFT detector [Lowe, 2004] to be among the best performing detectors for the application, which, however, is a costly operator that dominates the computation time of the GTBL Method. One approach is to use a Graphics Processing Unit (GPU) for feature extraction, but for the application in low-cost robots it is desirable to avoid the use of such dedicated hardware accelerators.

We hypothesize, that for ground texture based localization with available prior, proper keypoint detection can be disregarded. Instead, keypoints can be sampled randomly, mainly because pose estimation with a downward-facing camera is with good approximation a 2D problem. The camera pose can be estimated as an Euclidean transformation of rotation and translation in two dimensions. Therefore, keypoint object scale is constant for corresponding image regions in different images; and with an available pose prior, we can use its orientation in the map coordinate system as keypoint orientation, reducing the keypoint object properties that have to be determined to its image coordinates. Accordingly, for keypoint extraction in ground texture based localization with available prior, we are left with determining only the two degrees of freedom of translation, which leads us to the assumption that this is a point where we can save computational effort.

The contribution of this work is twofold. First, we show that keypoint sampling is a valid alternative to keypoint detection for ground texture based localization with an approximately known camera pose. This is shown as, for the three state-of-the-art feature-based localization methods Ranger [Kozak and Alban, 2016], StreetMap [Chen et al., 2018], and the GTBL Method [Schmid et al., 2020a], we achieve comparable success rates using sampled keypoint objects. This suggests that it is not necessary to recognize prominent features of the texture, but that any image patch presents sufficiently unique texture details. Second, we present *GTBL RND*, which stands for Ground Texture Based Localization (GTBL) method with RaNDomly sampled keypoints. GTBL RND is an adaptation of the GTBL Method, which uses keypoint sampling, achieves a success rate of more than 90% on the Micro-GPS database, and that takes roughly half as long as the next fastest method to compute.

This chapter is organized as follows. Section 8.1 introduces approaches to keypoint detection that focus on fast computation time, also we present existing work in the area of keypoint sampling. In Section 8.2, we introduce two novel performance metrics for local visual features, the *feature repeatability* and the *descriptor repeatability*, both aiming to capture the capability of a whole feature extraction pipeline of determining similar features for corresponding places in the overlap of independently recorded images. Section 8.3 introduces our keypoint sampling strategy, which is then evaluated in Section 8.4.1, using the Micro-GPS database, and in Section 8.4.2 using our HD Ground Database [Schmid et al., 2022]. Finally, Section 8.5 summarizes and discusses the obtained results.

8.1 Related Work

We review fast-to-compute detectors and existing work on keypoint sampling.

8.1.1 Speed-Optimized Keypoint Detection

Some keypoint detectors are designed with computation speed in mind. Among the fastest ones is FAST [Rosten and Drummond, 2006], a corner detector that considers a pixel to be part of a corner if multiple contiguous surrounding pixels are significantly darker or brighter. Computation speed of the method is improved by rejecting a pixel early after only a few comparisons, if the condition can no longer be fulfilled. Another fast-to-compute corner detector is Good Features To Track (GFTT) [Shi and Tomasi, 1994]. It simplifies the corner-score function of the Harris detector, which is based on an approximation of the local intensity change.

A second type of detectors use image pyramids to find scale-invariant keypoint objects. One of the most successful ones is SIFT [Lowe, 2004], detecting blobs as local intensity extrema in a DoG pyramid. SURF [Bay et al., 2006] and CenSurE [Agrawal et al., 2008] approximate the DoG, using the faster to compute Difference-of-Boxes or Difference-of-Octagons.

8.1.2 Keypoint Sampling

Keypoint detection typically identifies image regions that fulfill a keypoint criterion or that maximize a keypoint score function. *Keypoint sampling*, on the other hand, determines keypoint objects independent of the image content.

Keypoints can be sampled uniformly [Chatoux et al., 2016, van de Sande et al., 2010] or randomly [Nowak et al., 2006, Maree et al., 2005]. It has been used for image understanding tasks, where it is not necessary to retrieve corresponding image regions.

Tuytelaars [2010] developed a detector that starts with uniformly sampled keypoints, but improves the keypoint repeatability, moving keypoint objects to local optima of an *interestingness* measure.

Methods like DAISY [Tola et al., 2010], and SIFT flow [Liu et al., 2008] are used to compute feature descriptors at every position of a uniform grid. They can be used to find dense correspondences between image pairs, e. g. to compute optical flow.

8.2 Feature and Descriptor Repeatability

Keypoint repeatability was introduced by Mikolajczyk et al. [2005] as a quality measure of keypoint extractors. For two overlapping images, it measures the share of extracted keypoint objects that have been found in both images.

Whether keypoint objects are suited for a task also depends on the employed method for feature description and the matching strategy. Therefore, we introduce *feature repeatability*, which considers the whole feature instead of only the keypoint object. Feature repeatability is computed as the share of features from the overlap of two images that are correctly matched with each other. As a reminder: our keypoint objects specify both a position and an orientation; therefore, each match of features corresponds to a query image pose estimate. So, we employ the strategy that we introduced in Section 6.3.2, of considering matches to be correct if their corresponding pose estimates are correct, without taking the orientation error into account.

Another quality of interest for our method is what we refer to as *descriptor repeatability*. It measures the ratio of corresponding keypoint objects that are evaluated to the same feature descriptor. To evaluate this, we determine corresponding keypoint objects by projecting keypoint objects from an image a into an overlapping image b, and compare the descriptors assigned to them in the two images.

8.3 Method

We introduce GTBL RND. In order to be able to perform map-based localization, it has to solve the Problems 2 and 3 of map creation and localization. We build on the GTBL Method, but keypoint objects are not detected with SIFT, they are sampled randomly or uniformly. For the remainder of this dissertation, we refer to the variant of the GTBL Method using SIFT keypoint objects as *GTBL SIFT*, while we call this variant using (RaNDomly) sampled keypoints *GTBL RND*. We consider two possible keypoint sampling strategies: random sampling, and uniform sampling on a grid. Figure 8.1 presents SIFT, random, and uniformly sampled keypoint objects. Randomly sampled keypoints are retrieved as a simple random sample of image coordinates without replacement, which has greater computational cost than uniform sampling. However, using uniform sampling for query and reference images could lead to a situation where all keypoint objects are misaligned. Therefore, we extract randomly

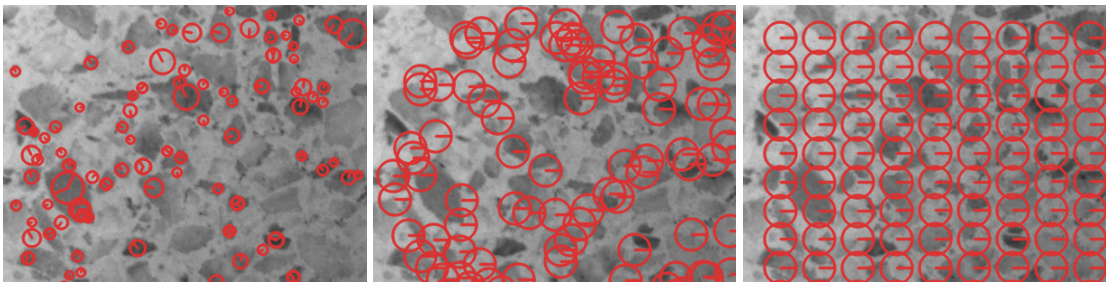


Figure 8.1: Illustration of keypoint objects extracted with (from left to right) SIFT, random keypoint sampling, and uniform keypoint sampling. Each keypoint object is visualized as a red circle. Keypoint object size is represented based the size of the circle, while its orientation is represented by a line from the center of the circle to its border. This illustration uses an image of the *tiles* dataset from the Micro-GPS database [Zhang et al., 2019]. This figure is taken from [Schmid et al., 2020b].

sampled keypoint objects from the reference images and uniformly sampled keypoint objects from query images, as computation time is not critical at map construction time, but only at localization time. Compared to a setting in which we also use randomly sampled keypoint objects for query images, we do not observe an effect onto the success rate.

In addition to the keypoint, we also need to define the scale, respectively the image patch size, and the orientation of the sampled keypoint objects. Here, we fix the scale, which is possible due to our assumption of a constant camera height. Further, we use the relative camera orientation to the map coordinate system as keypoint object orientation, which is known for the reference images and approximately known for the query images by using the orientation of the prior pose estimate.

Extensions: Three extensions to our method are evaluated.

1. *Repeatability constraint* for reference features (illustrated in Figure 8.2): only reference features with repeatable descriptors are stored during mapping. This can be done, if there are overlapping reference images. We project keypoint objects into the overlapping image to check if they are stable, i. e. if they are evaluated to the same descriptor, only then features are stored.

2. ***Multi-Map** (MM) approach*: for the reference images multiple sets of features are sampled. Each of them can be used during the localization step. This is similar to having multiple internal representations of the map, i. e. we repeat the map creation process of Problem 2 multiple times. Hence, we call this the Multi-Map (MM) approach. It means that for each

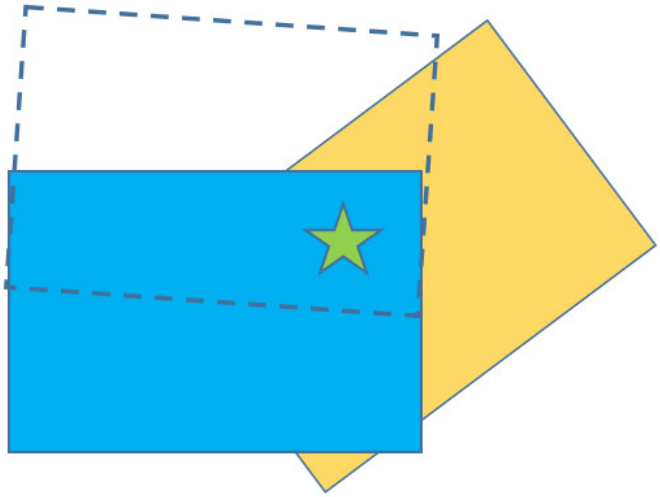


Figure 8.2: Illustration of the repeatability constraint: a feature (green star) from the reference image (blue rectangle) holds the repeatability constraint, if there is a (corresponding) keypoint object at the same place in an overlapping reference image (dashed rectangle), which is evaluated to the same feature descriptor. In comparison to a feature from a reference image that does not hold the repeatability constraint, we expect a reference image feature that does hold the repeatability constraint to possess a higher chance of having corresponding keypoint objects in overlapping query images (yellow rectangle) also to be evaluated to the same feature descriptor. This figure is taken from [Schmid et al., 2020b].

query image we perform multiple independent localization attempts (one for each map). The final pose estimate is determined by the localization attempt with most RANSAC inliers. Due to the use of random keypoint sampling each map will store a different set of features.

3. ***M**ulti-**M**ap approach with varying **O**rientations (MMO)*: this is similar to the MM approach, but for each map we apply a slightly different orientation to the keypoint objects (default is to use the known, *ground truth*, orientation of the reference images). Using additional maps with deviating orientations (e. g. with $\pm 5\,^{\circ}$) increases the independence of the localization attempts and the robustness to orientation.

The employment of keypoint sampling for GTBL RND leads to poor keypoint repeatability, and consequently to poor feature repeatability. However, with more reference features, feature repeatability increases as can be seen in Figure 8.3. This why we increase the number of retrieved features of reference images from 850 for GTBL SIFT to 5000 for GTBL RND. Similarly, we can increase the number of correct feature matches, if we consider a larger number of query image features. Here, we increase the number from 850 for GTBL SIFT to 2000. These

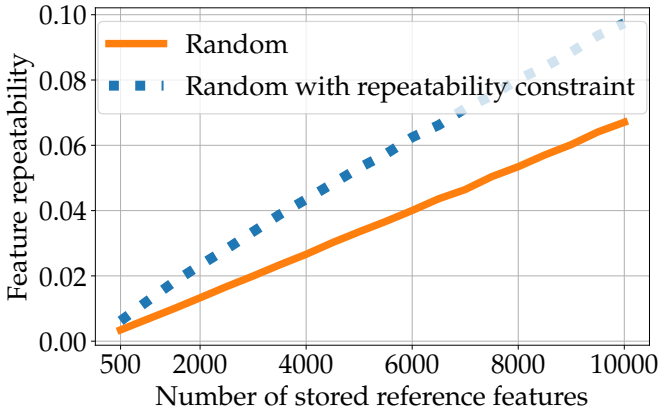


Figure 8.3: Mean feature repeatability on the Micro-GPS database [Zhang et al., 2019] for varying numbers of stored features per reference image, using random keypoint sampling, with and without applied repeatability constraint, and using identity matching with 15-bit LATCH [Levi and Hassner, 2016] descriptors as in the GTBL Method [Schmid et al., 2020a]. This figure is adapted from [Schmid et al., 2020b].

values are a trade-off between localization success rate and computation time (and memory consumption). We observe that the success rate of GTBL RND is starting to saturate at these values. Accordingly, we find larger number not to be worthwhile. Figure 8.3 also shows, that the employment of the repeatability constraint further improves the feature repeatability. In practice, we could filter keypoint objects based on the repeatability constraint in an offline mapping phase, which would allow us to reduce the number of stored features, reducing the computation effort and memory consumption of GTBL RND in the online localization phase.

8.4 Evaluation

In our evaluation, we examine the task of absolute map-based localization with available prior.

As in Chapter 5, in order to generate the prior pose estimates, we take the ground truth poses, shift them with a fixed distance d into a randomly sampled direction, and rotate it with an orientation sampled from a zero-mean normal distribution. In our evaluations, reference images for localization are chosen based on proximity to the prior position estimate using a k-d-tree.

In Section 8.4.1, we evaluate the proposed method extensively on the Micro-GPS database of Zhang et al. [2019], also considering the proposed extensions, and faster-to-compute keypoint detectors as an alternative to keypoint sampling.

Afterwards, in Section 8.4.2, we compare GTBL RND in its default configuration with GTBL SIFT, Ranger, and StreetMap on the HD Ground Database [Schmid et al., 2022].

Our main performance metric is the localization success rate. Again, we adopt the error thresholds of Zhang et al. [2019]: a localization attempt is considered to be correct if its translation error to the ground truth pose is smaller than 4.8 mm and if the absolute orientation error is smaller than 1.5 °.

The experiments are evaluated with an E3-1270 Intel Xeon CPU.

8.4.1 Evaluation on the Micro-GPS Database

This section presents the evaluation published in [Schmid et al., 2020b].

We evaluate on the six ground textures (fine and coarse asphalt, carpet, concrete, tiles, and wood) from the PointGrey CM3 database of Zhang et al. [2019].

Besides random and uniform keypoint sampling, we examine SIFT and SURF, as well as the three detectors, which we found in Chapter 4 to be the fastest among the implemented methods in OpenCV 4.0 [Bradski, 2000]: FAST, GFTT, and CenSurE. Since these detectors do not provide keypoint object orientation, we apply the same strategy as for keypoint sampling, using the orientation of the localization prior.

In addition to our localization methods, we examine Ranger [Kozak and Alban, 2016] and StreetMap [Chen et al., 2018]. We re-implemented both, using the details provided by the authors. All methods are evaluated with the same query and reference images, and the same random seeds to generate the localization prior. For all series of experiments involving random keypoint sampling, results are averaged over three repetitions. In the following, we will mention the corresponding keypoint extraction method after the localization method, using the abbreviation *RND* for our keypoint sampling strategy that combines random and uniform keypoint sampling, e. g. Ranger FAST, StreetMap SURF, and StreetMap RND.

How many reference images are considered during localization attempts depends on the shift distance d, i. e. the position prior accuracy. We employ the same numbers as for our previous experiments on the Micro-GPS database, presented in Section 5.2.1.2. They range from 5 images for experiments with zero shift distance, over 20 images for distances of 0.1 m, to 250 images for 0.5 m.

8.4.1.1 Implementations

For GTBL SIFT and StreetMap, we use the same implementations and parameters as presented in Section 5.2.1.1, which have been optimized on the Micro-GPS database for localization success rate and computation speed. Parameters of FAST, GFTT, and CenSurE are taken from the feature survey of Chapter 4, presented in the appendix Section A.1, where they were optimized for keypoint repeatability and computation speed.

The implementation of GTBL RND is the same as for GTBL SIFT, but SIFT keypoint detection is exchanged with our keypoint sampling strategy, and the number of extracted features is increased. As previously mentioned, we store 5000 features per reference image, and use 2000 feature per query image for feature matching.

For Ranger, we previously employed AKAZE, instead of using CenSurE, as proposed by Kozak and Alban [2016]. This was due to CenSurE not providing keypoint object orientations. In this evaluation, again making use of the aforementioned strategy of using the orientation of the prior as keypoint object orientation, we implement Ranger with BRIEF descriptors computed on CenSurE keypoint objects. Our Ranger implementation extracts 1250 CenSurE keypoint objects with a maximum patch size of 14, a response threshold of 0, a projected line threshold value of 29, a binarized threshold value of 22, and a non-maximum suppression size of 2. Keypoint objects are described using the rotation invariant 64-byte BRIEF descriptor. Otherwise, the implementation is the same as described in Section 5.2.1.1.

We also evaluate StreetMap and Ranger using keypoint sampling. Here, we have to define the number of sampled keypoint objects. This parameter is optimized again for localization success rate (primary) and computation time (secondary), using the same set of training images used in Chapter 5, which is a separate set of query images to those used for the performance evaluation.

8.4.1.2 Results: Descriptor and Feature Repeatability

Descriptor and feature repeatability are examined for different keypoint extraction approaches. The evaluation is done for our approach of using identity matching with compact binary descriptors, i. e. the first 15-bit of LATCH.

Additionally, the repeatability constraint (see Figure 8.2) is examined. For this purpose, overlapping reference images are required, but the provided reference images in the Micro-GPS database are not significantly overlapping. Therefore,

Table 8.1: Mean **descriptor repeatability** values, i. e. the ratio of corresponding keypoint objects that have been evaluated to the bit-identical descriptor value in overlapping images, for varying keypoint extraction methods without and with applied repeatability constraint. Corresponding keypoint objects are generated by projecting the extracted keypoint objects of one image into the other. The 15-bit LATCH [Levi and Hassner, 2016] descriptor of the GTBL Method [Schmid et al., 2020a] is used to determine feature descriptor values. The results of the descriptor repeatability metric have been averaged over the evaluated textures of the Micro-GPS database [Zhang et al., 2019].

Repeatability constraint	Random	Uniform	FAST	GFTT	CenSurE	SIFT	SURF
Without	4.2%	4.0%	5.0%	5.0%	5.1%	4.6%	6.1%
With	7.1%	7.0%	7.5%	8.0%	7.0%	6.3%	7.4%

Table 8.2: Mean **feature repeatability** values, i. e. the ratio of features, which have been extracted independently for two overlapping images, that are correctly matched with each other, for varying keypoint extraction methods without and with applied repeatability constraint. As in the GTBL Method [Schmid et al., 2020a], we employ the 15-bit LATCH [Levi and Hassner, 2016] descriptor and identity matching to determine feature matches. Two different approaches to feature selection are evaluated to determine subsets of 1000 keypoints objects per image: the Non-Maximum Suppression (NMS) technique and simple random selection. The values have been averaged over the evaluated textures of the Micro-GPS database [Zhang et al., 2019].

Keypoint selection	Repeatability constraint	Random	Uniform	FAST	GFTT	CenSurE	SIFT	SURF
NMS	Without	0.7%[1]	0.7%[1]	8.0%	9.8%	12.7%	13.0%	15.6%
NMS	With	1.2%[1]	1.3%[1]	1.7%	4.4%	3.0%	6.3%	1.0%
Random	Without	0.7%	0.7%	0.7%	9.8%	1.2%	13.0%	1.3%
Random	With	1.2%	1.3%	1.3%	5.4%	2.3%	6.3%	2.5%

we use the image sequences intended for localization, which have significant intersection area (22.7% on average), as reference images, and images intended for mapping as query images. The results presented in Table 8.1 and 8.2 are averaged over 600 image pairs (100 per texture type) of reference and query image pairs with an intersection of at least 20%. From each image of these pairs up to 1000 features, satisfying the constraint, are extracted.

Our results show that the descriptor repeatability is similar for all evaluated keypoint extractors. Thus, the choice of the keypoint extractor does not have a strong effect on the probability of evaluating corresponding keypoint objects to the same 15-bit LATCH descriptor. Additionally, when storing only features

[1] Keypoint score not available; therefore, equivalent to random selection.

that meet the repeatability constraint, descriptor repeatability increases for all keypoint extractors.

The results (Table 8.2 first row) confirm our previous findings from Chapter 4 that SIFT, SURF, and CenSurE are among the best keypoint detectors for ground texture images. Descriptor repeatability is similar for all keypoint extraction methods. Accordingly, variances in feature repeatability result from variances in keypoint repeatability. This explains the poor performance of randomly and uniformly sampled keypoint objects. If we increase the number of sampled keypoint objects from 1000 to 5000, which is the number we will use for reference images in the following evaluation of localization performance, the feature repeatability of randomly sampled keypoint objects improves from 0.7% to 3.3%. Furthermore, the repeatability constraint increases feature repeatability of sampled keypoints (Table 8.2 second row). Again, using 5000 instead of 1000 reference features further improves the result (from 1.2% to 5.3% for randomly sampled keypoint objects).

Interestingly to us, the feature repeatability of proper keypoint detectors worsens with applied repeatability constraint. This can be explained with the keypoint selection method. The evaluated keypoint detectors provide a keypoint score, which is used to select the *best* 1000 features per image with NMS. Selecting a subset of the detected keypoints randomly instead, decreases the feature repeatability (Table 8.2 third row). This is not the case for GFTT and SIFT, because, in contrast to the other methods, they find less than 1000 keypoint objects on average (SIFT 392.5, GFTT 933.9). If we apply the repeatability constraint to randomly selected keypoint objects, the feature repeatability is increased, as for randomly and uniformly sampled ones (Table 8.2 fourth row). Again, this is not the case for GFTT and SIFT, possibly because their already small number of extracted features is further decreased with applied repeatability constraint. Overall, only sampled keypoint objects benefit from the repeatability constraint. For keypoint detectors it is better to rely on score based selection.

8.4.1.3 Results: Localization Performance

For the following experiments, we evaluated 500 test sequence images per texture type. Figure 8.4 shows localization success rates for our method with different numbers of query features and with increasing position prior error, the applied orientation prior has a standard deviation of 5.0 °. We notice, using more than our default value of 2000 query features does not increase performance by much. A prior with translation error of up to 0.2 m decreases the

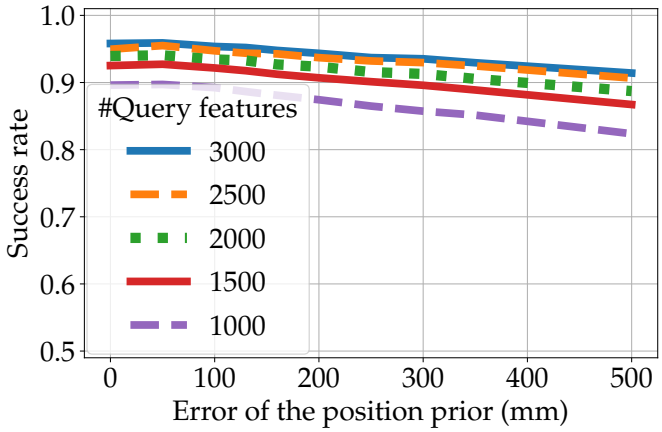


Figure 8.4: Success rates of GTBL RND, i. e. the GTBL Method [Schmid et al., 2020a] using random keypoint sampling (RND) instead of SIFT [Lowe, 2004] for keypoint object extraction, with varying numbers of query image features, evaluated on the Micro-GPS database [Zhang et al., 2019] for varying position prior accuracies. This figure is adapted from [Schmid et al., 2020b].

success rate of our method only slightly: with 2000 query features it decreases from 94.1% without error to 92.2% with an error of 0.2 m. For larger translation errors, it decreases further (88.7% for an error of 0.5 m). Success rates of Ranger and StreetMap are less affected by the translation error, with an error of 0.5 m they still reach success rates of 99.8% and 97.6%, but Figure 8.5 shows that they become slow with larger numbers of considered reference images, which can be explained by the use of nearest neighbor matching. Ranger is particularly fast for small translation errors (and therefore small numbers of considered reference images), because it terminates as soon as a well matching reference image is found. With a translation error of 0.1 m Ranger requires 14.7 ms and StreetMap 38.8 ms for the feature matching process, with an error of 0.2 m this increases to 47.8 ms and 96.5 ms. For our method this increases matching time only slightly from 5.0 ms to 11.6 ms. In the following, we fix the position prior error to 0.1 m.

A key parameter for localization with keypoint sampling is the number of features stored per reference image. For Ranger, we find that with 750 sampled keypoint objects it already reaches a similar success rate as with CenSurE (99.6% to 99.9%). StreetMap benefits from larger numbers of reference features; with 2000 sampled keypoint objects it reaches 99.1% success rate. Using more keypoint objects improves performance only slightly, but significantly increases computational cost. Our method, due to the use of identity matching, requires more reference features to reach good performance. Here, we extract 5000 features per reference image.

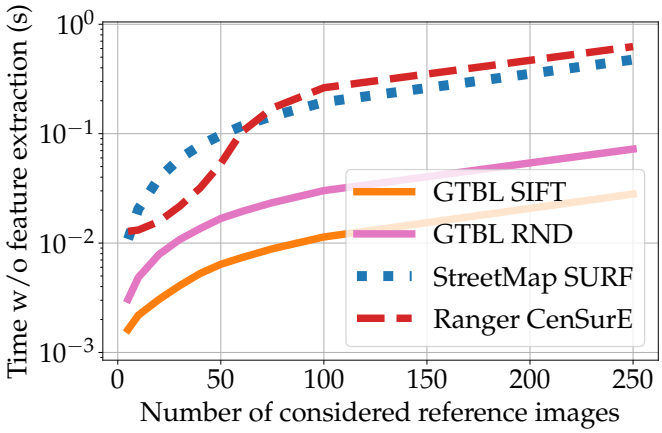


Figure 8.5: Accumulated computation time during localization (without the time required for keypoint detection and feature description) of StreetMap [Chen et al., 2018], Ranger [Kozak and Alban, 2016], and the GTBL Method [Schmid et al., 2020a], using keypoint objects extracted with SIFT [Lowe, 2004] and random keypoint sampling (RND), evaluated on the Micro-GPS database [Zhang et al., 2019] for varying numbers of considered reference images, based on varying position prior accuracies. This figure is adapted from [Schmid et al., 2020b].

The evaluation of the localization success rate for varying orientation prior standard deviations (Figure 8.6) demonstrates the limits of our method with sampled keypoint objects. The method relies on corresponding keypoint objects being evaluated to the same descriptor, which requires a good estimate of keypoint object orientation. StreetMap is not affected by the orientation prior, because during feature description SURF estimates keypoint orientation itself. The success rates of Ranger RND are significantly decreased only for large errors in the orientation prior beyond 7.5 °.

The following experiments are evaluated with an orientation prior with a standard deviation of 5.0 °. We evaluate the localization methods, using their default keypoint detectors (SURF for StreetMap, CenSurE for Ranger, and SIFT for GTBL SIFT) the fast-to-compute detectors CenSurE, FAST, and GFTT, and keypoint sampling. Table 8.3 presents the results.

Ranger is the most successful method with 99.9% success rate using CenSurE or FAST keypoint objects. Our method with keypoint sampling is with 25.4 ms the fastest method, taking less than half as long as Ranger using either CenSurE or FAST, but with 93.5% it has a lower success rate. However, closer analysis shows that GTBL RND has a success rate of 99.8% if the sampled error of the orientation prior is smaller than 4 °. The time required for the final pose estimation step increases for our method when using keypoint sampling because

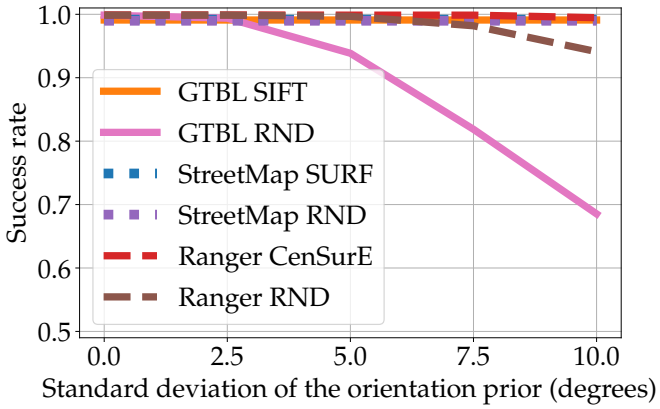


Figure 8.6: Success rates evaluated on the Micro-GPS database [Zhang et al., 2019] for increasing standard deviation of the orientation prior. The GTBL Method [Schmid et al., 2020a], StreetMap [Chen et al., 2018], and Ranger [Kozak and Alban, 2016] are evaluated with their respective default approach for keypoint extraction (SIFT [Lowe, 2004], SURF [Bay et al., 2006], and CenSurE [Agrawal et al., 2008]), as well as with random keypoint sampling (RND). This figure is adapted from [Schmid et al., 2020b].

more matches participate in the voting procedure. Ranger and StreetMap reach similar success rates with keypoint sampling as with their default detector (CenSurE for Ranger and SURF for StreetMap), but it increases their required time for feature matching significantly, mainly due to the use of nearest neighbor matching. This is particularly problematic for Ranger, because it matches with multiple reference images individually. For StreetMap, the employment of our random keypoint sampling strategy is a suitable alternative, as it has almost the same success rate as with SURF keypoint objects, but is overall about 35 ms faster to compute. Using the faster to compute keypoint detectors for StreetMap decreases computation time even further, but it also decreases the success rate.

Evaluation of the Multi-Map Approach: We evaluate the previously proposed Multi-Map (MM) approach and the Multi-Map with varying Orientations (MMO) approach. In both cases, multiple sets of features are independently sampled for each reference image. This is analogous to having multiple copies of each reference image, being treated independently of each other, respectively having multiple maps of the same application area. For each available map, a localization attempt is performed for the query image. The attempt for which we observe the most RANSAC inliers is used as final pose estimate. In the MM approach only the keypoint object positions vary for the different features sets of the same reference image, while the keypoint object orientation is the same (namely the orientation of the prior). In the MMO approach, a slightly different

Table 8.3: Evaluation of the success rate and task-specific computation times for varying keypoint detectors employed for Ranger [Kozak and Alban, 2016], StreetMap [Chen et al., 2018], and the GTBL Method [Schmid et al., 2020a], for which we also test the multi-map approach (MMO-2 and MMO-4). Localization is performed with a given prior that has a 0.1 m position error and a standard deviation of the orientation error of 5.0 °. Results are averaged for the evaluated textures of the Micro-GPS database [Zhang et al., 2019]. The best success rates and overall computation time are highlighted in bold.

Method	Keypoint detector	Success rate	Computation time (ms) Overall	Detection	Desc.	Matching	Pose est.
Ranger	CenSurE	**99.9**%	55.7	30.4	9.5	14.7	< 0.1
Ranger	FAST	**99.9**%	61.8	14.1	9.2	34.0	4.1
Ranger	GFTT	95.0%	46.2	22.8	8.1	13.2	1.7
Ranger	RND	99.6%	187.6	< 0.1	9.2	147.4	30.6
StreetMap	SURF	99.2%	153.3	95.6	18.4	38.8	< 0.1
StreetMap	CenSurE	83.7%	74.2	28.6	4.4	40.3	< 0.1
StreetMap	FAST	63.8%	59.5	1.5	4.6	3.8	1.2
StreetMap	GFTT	43.7%	62.1	22.6	3.5	34.5	1.3
StreetMap	RND	99.1%	116.9	< 0.1	21.4	94.7	< 0.1
GTBL	SIFT	99.1%	730.1	716.9	10.1	2.1	< 0.1
GTBL	CenSurE	95.2%	40.4	29.0	8.3	2.3	< 0.1
GTBL	FAST	88.1%	25.7	14.7	7.8	2.5	< 0.1
GTBL	GFTT	85.9%	32.6	22.5	7.3	2.2	< 0.1
GTBL	RND	93.5%	**25.4**	< 0.1	17.5	5.0	2.3
GTBL MMO-2	RND	97.9%	33.2	< 0.1	18.0	10.4	4.7
GTBL MMO-4	RND	99.5%	47.9	< 0.1	17.8	20.6	9.4

keypoint object orientation is applied for each feature set.

Using two maps with the MM approach increases the success rate to 95.1% (from 93.5% with a single map), with four maps it increases to 95.6%.

The multi-map approach with applied orientation deviations further improves performance. Using two maps with orientation deviations of $\{-2.5\,^\circ, 2.5\,^\circ\}$ increases the success rate to 97.9%, with approximately 8 ms longer computing time. (GTBL MMO-2 in Table 8.3). With four maps and orientation deviations of $\{-6.0\,^\circ, -2, 0\,^\circ, 2.0\,^\circ, 6.0\,^\circ\}$ (GTBL MMO-4 in Table 8.3), the method has a success rate of 99.5%, still being roughly 8 ms faster to compute than Ranger with CenSurE.

An additional advantage of the MM approach is that with multiple localization attempts we can perform a consistency check, requiring multiple pose estimates to confirm each other. If we require that at least two of the pose estimates are close to each other (closer than 4.8 mm and with less than 1.5 ° orientation difference), we reject 63 out of 64 unsuccessful localization attempts for MMO-2, while also rejecting 14.0% (412) of successful attempts. With MMO-4 this leads

to a rejection of 11 of 14 unsuccessful attempts, and 1.84% (55) successful ones.

An alternative to the MM and MMO approaches of storing multiple maps with independently sampled features, would be to simply store more features per reference images. However, storing more than 5000 features per reference image does not significantly increase performance. The advantage of the multi-map approach lies in its ability to perform multiple *independent* localization attempts.

8.4.2 Evaluation on the HD Ground Database

This section presents the evaluation published in [Schmid et al., 2022], examining the proposed GTBL RND method on the HD Ground Database.

We evaluate the localization performance of Ranger [Kozak and Alban, 2016] in its default variant using CenSurE keypoint objects, StreetMap [Chen et al., 2018] in its default variant using SURF features, GTBL SIFT [Schmid et al., 2020a], and GTBL RND [Schmid et al., 2020b]. Again, we provide prior pose estimates, which are generated by taking the ground truth pose of the query image, translating it with a specified distance d into a randomly sampled direction and rotating it with an orientation angle sampled from a zero-mean normal distribution. As explained in Section 5.2.2.2, we obtain the number of closest reference images that are considered during localization as

$$\left\lceil \frac{\pi d^2}{0.12\,\mathrm{m} \cdot 0.16\,\mathrm{m}} \right\rceil \cdot 9. \tag{8.1}$$

For each texture, evaluation is done on the two regular test sequences that were recorded with shortest time distance to the time of map creation.

8.4.2.1 Implementation

We implement the methods as described in Chapter 5. Additionally, we find texture-specific parameter settings for GTBL RND, using the same parameter optimization strategy. The found values for image scale and the number of extracted query image features are presented in Table 8.4. The remaining parameters are presented in the appendix in Section A.5. In comparison with GTBL SIFT, the employed image scales are slightly smaller. This might be due to the circumstance that processing images at a lower resolution increases the likelihood of finding matching keypoint objects by chance with our keypoint sampling strategy.

Table 8.4: Texture-specific optimized values for image scale and the resulting distance covered per pixel in millimeters, as well as the number of features per image for GTBL RND, on the HD Ground Database [Schmid et al., 2022].

Texture	Approach	Image scale	mm/pixel	#Features
Aphalt	GTBL RND	0.20	0.50	3500
Cobblestone	GTBL RND	0.23	0.43	2200
Carpet	GTBL RND	0.23	0.43	2100
Laminate	GTBL RND	0.15	0.67	2300

In addition to the texture-specific parameter settings, that we obtained in previous chapters, we find a set of generalized parameters for each localization method, optimizing jointly on all training areas of the HD Ground Database. The values are presented in the respective sections of the appendix in Chapter A. We use the generalized parameter settings to examine the generalization capabilities, respectively the sensitivity to parametrization, by comparing the results that we obtain with the texture-specific parameter settings with those that we obtain with the generalized parameter settings.

8.4.2.2 Results

Our first two experiments are evaluated on the main textures with texture-specific parametrization, using the two regular test sequences with shortest time interval to mapping. Their results are presented in Figure 8.7.

Our first experiment is similar to the one of Section 5.2.2.2. We evaluate with the SD of the prior being fixed to 3.0 ° while the position prior accuracy d varies between 50 and 2000 mm. Again, we observe that StreetMap achieves very high success rates, independently of the position prior accuracy, while for Ranger, success rates decrease slowly for d values above 250 mm. GTBL RND achieves slightly lower success rates than GTBL SIFT.

We observe for small numbers of considered reference images that the overall computation time is dominated by the required time for feature extraction, while it is dominated by the time for feature matching for larger numbers of considered reference images. This is why, GTBL SIFT is slow for accurate position priors. For less accurate position priors, the GTBL methods have an advantage using the identity matching technique instead of nearest neighbor feature matching. But, GTBL RND scales not as favorable as GTBL SIFT with larger position prior inaccuracies, due to the large number of considered features per reference image. However, because of the employment of keypoint

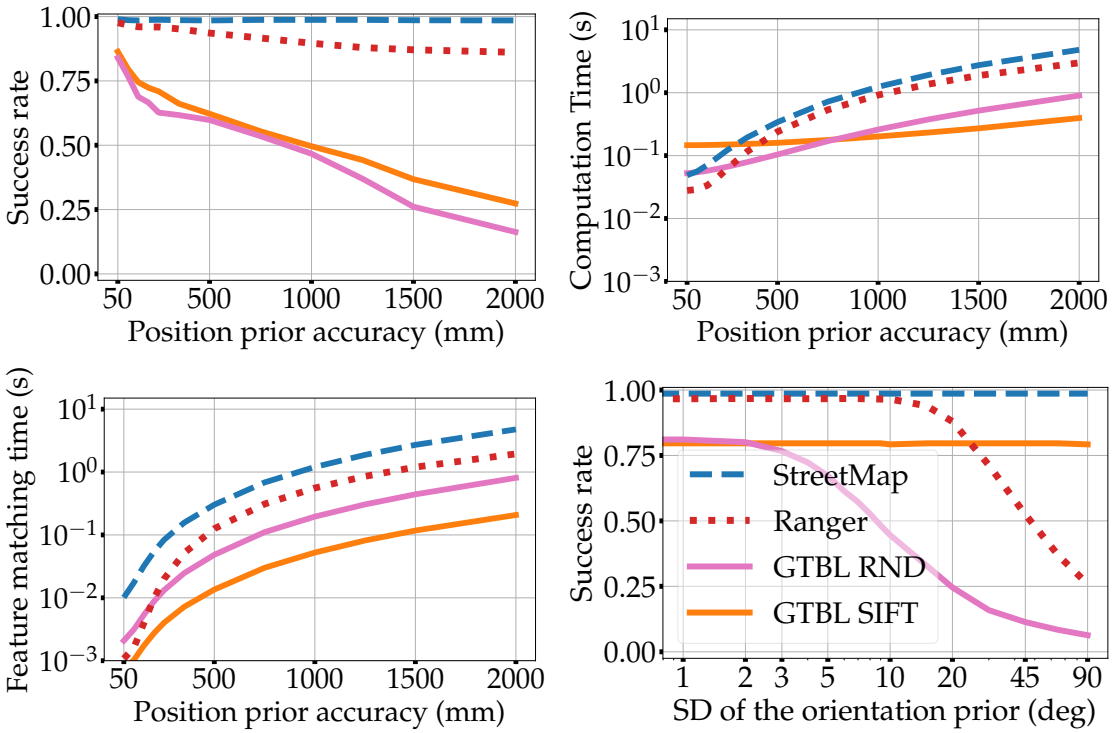


Figure 8.7: Local localization performance of StreetMap [Chen et al., 2018], Ranger [Kozak and Alban, 2016], and the GTBL Method [Schmid et al., 2020a] using random keypoint sampling (RND) and SIFT [Lowe, 2004] for keypoint extraction, averaged over the four main textures of the HD Ground Database [Schmid et al., 2022]. The top left plot presents the success rate for varying position prior accuracies; the top right one presents the overall localization computation time for varying position prior accuracies; the bottom left one presents the part of the computation time that is used for feature matching for varying position prior accuracies; and the bottom right one presents the success rate for varying standard deviation of the orientation prior. This figure is adapted from [Schmid et al., 2022].

sampling, the computation times for more accurate position priors is reduced.

In a second experiment (see Figure 8.7, bottom right), we fix the position prior accuracy to $d = 100\,\text{mm}$ and vary the orientation prior SD between $1\,^\circ$ and $90\,^\circ$. StreetMap and GTBL SIFT determine feature orientations on their own and are not affected by this. But Ranger and GTBL RND use the orientation prior as orientation of the query features. Again, we find Ranger's use of BRIEF descriptors with nearest neighbor matching to be more robust to rotation than GTBL RND's 15-bit LATCH with identity matching.

Generalization Capabilities In another experiment, we assess the generalization capabilities of the methods, using the jointly optimized parameters. Table 8.5 presents the success rate on the main textures as well as on the six

Table 8.5: **Local localization**, i. e. localization with a given prior pose estimate, success rates of Ranger [Kozak and Alban, 2016], StreetMap [Chen et al., 2018], and the GTBL Method [Schmid et al., 2020a] using SIFT [Lowe, 2004] and random keypoint sampling (RND) for keypoint extraction, evaluated on the main and generalization textures of the HD Ground Database [Schmid et al., 2022] with the jointly optimized parameters.

Texture type	Ranger	StreetMap	GTBL SIFT	GTBL RND
Main	0.964	0.986	0.888	0.696
Generalization	0.988	0.898	0.700	0.528

Table 8.6: **Relative localization**, i. e. incremental localization from one image to a consecutively recorded one, success rates of Ranger [Kozak and Alban, 2016], StreetMap [Chen et al., 2018], and the GTBL Method [Schmid et al., 2020a] using SIFT [Lowe, 2004] and random keypoint sampling (RND) for keypoint extraction, evaluated on the main and generalization textures of the HD Ground Database [Schmid et al., 2022] with the jointly optimized parameters.

Texture type	Ranger	StreetMap	GTBL SIFT	GTBL RND
Main	0.947	0.948	0.830	0.926
Generalization	0.976	0.956	0.485	0.624

generalization textures with a position prior accuracy of 100 mm and an orientation prior SD of 3 °. For comparison, the corresponding average success rates on the main textures using the texture-specific parameters are 0.961 for Ranger, 0.986 for StreetMap, 0.775 for GTBL SIFT, and 0.716 for GTBL RND. This means that the performance is similar for Ranger, StreetMap, and GTBL RND, using the jointly optimized parameters, while it improved significantly for GTBL SIFT, which could mean that its texture-specific parametrization overfitted to the training areas. For Ranger, we observe generally very high success rates and a better performance on the generalization textures as on the main textures, while it is the other way around for the other three localization methods, suggesting that Ranger has the best generalization capabilities.

Relative Localization The same methods, used for localization with available prior, can also be employed for relative localization. Here, query image poses are estimated in respect to their predecessor image of the sequence, which is projected onto its ground truth position. Therefore, we can evaluate the localization success rate, and, since we know the ground truth poses of the query images, also the translation and orientation errors. We observe an average

movement between consecutive images of 92.2 mm and 3.1 °. Success rates are presented in Table 8.6.

Again, the GTBL methods perform better on the main textures than on the generalization textures. For this task, GTBL RND has higher success rates to what we observed for local localization, while the success rates of Ranger and StreetMap are similar. On the other hand, when comparing the average displacement errors of successful localization attempts, we observe with 0.77 mm a larger value for GTBL RND, than for StreetMap (0.34 mm), Ranger (0.38 mm), and GTBL SIFT (0.39 mm). Similarly, we observe a larger average orientation error for GTBL RND of 0.42 °, compared to StreetMap with 0.11 °, Ranger with 0.10 °, and GTBL SIFT with 0.17 °. GTBL RND tends to be less accurate than the other methods. This is probably an effect of the random keypoint sampling technique, where correct feature correspondences are created only by randomly selecting image patches that are sufficiently close to each other.

8.5 Discussion

In this chapter, we examined approaches to ground texture based localization with available prior pose estimate. For this purpose, we proposed keypoint sampling as an alternative to proper detection of keypoint objects.

We evaluated Ranger [Kozak and Alban, 2016], StreetMap [Chen et al., 2018], and the previously introduced GTBL Method [Schmid et al., 2020a], which we call GTBL SIFT if used with SIFT keypoint objects and GTBL RND if used with our keypoint sampling strategy. These methods can achieve similar localization success with detected and sampled keypoint objects. While StreetMap and the GTBL Method can benefit from this approach, Ranger is slowed down, due to the increased effort for feature matching. If the translation error of the available prior pose estimate is about 0.1 m or less, GTBL RND is significantly faster to compute than GTBL SIFT. However, we observe that this acceleration is paid for with slightly lower success rates.

Overall, it seems that keypoint sampling is a suitable alternative to proper keypoint detection for ground texture based localization if a sufficiently accurate prior approximation of the camera pose is available. In particular, the method requires orientation priors of about 4 ° or less. In practice, this can be realized with rapid localization updates, which are possible with the efficient combination of identity matching, compact binary descriptors, and keypoint sampling. Therefore, due to the availability of more accurate priors, the GTBL

Method with keypoint sampling is probably better in practice than in some of our experiments.

Still, considering the results on the more difficult HD Ground Database [Schmid et al., 2022], we found StreetMap, and in particular, Ranger to have a more reliable localization performance than both variants of the GTBL Method. With proper parametrization and a sufficiently accurate prior, they are similarly and sometimes even faster to compute.

9 Conclusion and Outlook

Section 9.1 brings together the findings of the presented studies of this dissertation. In Section 9.2, we will then discuss possible targets for future research.

9.1 Retrospection

We review the knowledge gain from this dissertation through an assessment of three questions.

What to look out for when equipping a robot for ground texture based localization? We studied this question in Chapter 3, where we presented our mobile robot platform that we used to scan the ground of four large-scale application areas. In this context we derived guidelines for the robot design, based on the supported vehicle speed, the desired overlap of consecutive images, the maximum allowed motion blur, as well as the employed camera and objective. This allowed us to find appropriate values for the exposure time and the camera height. Similarly, if the camera height is restricted, we could derive the required camera opening angle, or the required recording frequency, for example.

In this context, we observed the need for short exposure times, due to the fact that with a downward-facing camera, motion blur corresponds exactly to the traveled distance during exposure. This led to a significantly increased image quality for our recordings of the HD Ground Database, with a exposure time of 0.1 ms, in comparison to that of the Micro-GPS database with an exposure time of 3 to 5 ms.

A particular difficulty occurs in the case of reflective ground textures. With our robot setup, using an off-the-shelf LED ring for illumination, we observe specular reflections on some ground materials and on wet surfaces, rendering a significant part of the recorded images as useless for our task. Here, as explained in Section 3.3, we found cross-polarization to be a simple and effective solution. However, this technique requires to compensate for the lost illuminance. An alternative approach to cross-polarization is the employment of a

dome illumination, in which the ground is illuminated homogeneously from all directions by indirect lighting.

In addition to these findings on the preferred robot setup, consideration should be given to an appropriate course of action for the scanning of the ground. In particular, we found it to be helpful to record the data in a structured way, i. e. dividing the application areas into smaller areas, each of which is scanned lane by lane. Otherwise, without following a clear pattern during scanning, it might be difficult to cover the application area completely, and it might be tricky to identify the overlapping images for initial pose estimates.

During the map creation phase, it is highly desirable to avoid the inclusion of incorrect (feature) correspondences between the images, in particular if, as in our case, the squared reprojection error is minimized globally. For this purpose, we proposed to perform a consistency check for the estimated reference image pose, and to use only the RANSAC inliers of the feature-based pose estimation.

What are important trade-offs in the employment of ground texture based localization? In our survey of suitable keypoint detectors for ground texture based localization in Chapter 4, we observed that the detectors with particularly short computation times (CenSurE, GFTT, and FAST) do not provide keypoint object orientation information. Similarly, one of the main drawbacks of our strategy to avoid proper keypoint detection with keypoint sampling, that we proposed in Chapter 8, is the lack of keypoint object orientation information. This indicates a trade-off between the computational effort of keypoint object extraction and the availability of orientation information. However, missing the orientation information is not a problem for the task of relative localization, if the camera orientation of consecutive images changes only slightly, and it is not a problem for the task of map-based localization if a sufficiently accurate prior is available. In that case, as suggested and examined in Chapter 8, we can use the approximated camera orientation in the map coordinate system as keypoint object orientation. Accordingly, the suspected trade-off can be resolved depending on the use case of the localization method. Furthermore, shorter computation times for localization can result in a positive feedback loop if it reduces the time interval between consecutively localized images. In such a case, the available prior is more accurate, which can again result in a further reduced computation time for localization. In such a case, the employment of faster-to-compute methods for keypoint object extraction can even be beneficial to the accuracy of the available orientation information.

We observed another trade-off between the feature matching speed and the

possibility to exploit an available pose prior. This trade-off manifests in the decision whether to use nearest neighbor matching matching, as in StreetMap [Chen et al., 2018] and Ranger [Kozak and Alban, 2016], where each query image feature is compared with each reference image feature, and Approximate Nearest Neighbor (ANN) feature matching, where an efficient search structure for matching is build offline, as in Micro-GPS [Zhang et al., 2019]. While nearest neighbor matching allows to select the set of considered reference image features online at localization time, its computation time grows linearly with the size of that set. Having n_q query image features, n_r features per reference image, and m considered reference images, its processing time has a complexity of $\mathcal{O}(n_q \cdot n_r \cdot m)$, comparing each query feature with each reference feature. With a pre-constructed data structure, on the other hand, finding the nearest neighbor, or the Approximate Nearest Neighbor (ANN), of a descriptor is more efficient to compute, e. g. using a k-d tree it has an average complexity of $\mathcal{O}(\log(n_r \cdot m))$, resulting in a total average complexity for the matching process of $\mathcal{O}(n_q \cdot \log(n_r \cdot m))$. But this approach is not practical if we want to consider only those reference images that are close to our given prior. In Chapter 5, we proposed *identity matching* as a novel cost-effective solution. It has a complexity of $\mathcal{O}(n_q \cdot m)$ for feature matching, as it performs a single table look-up per query feature and reference image. Also, as in the case of classical nearest neighbor matching, identity matching allows to select the set of considered reference images online, based on spatial distance of the corresponding reference image pose to the prior. We found our GTBL Method based on identity matching to be well suited for the challenges of the Micro-GPS database [Zhang et al., 2019], for the more difficult HD Ground Database, however, we observed good performance only for accurate pose priors, or in the use case of a teach-and-repeat scenario, where the amount of considered reference images is significantly smaller. Also, we observed the GTBL Method to be highly dependent on the employed parametrization. For this challenge, we proposed in Chapter 6 a model-based prediction model of the localization success rate, and we showed that it allows to find suitable parameters automatically in short time. An alternative to the use of a prior pose estimate, is the employment of image retrieval. Similarly as with a prior pose estimate, image retrieval can be used to reduce the number of considered reference image features. The goal of both approaches is to consider only the features of reference images that have actual overlap with the query image. In Chapter 7, we proposed a Deep Metric Learning (DML) approach that retrieves the overlapping images with a recall of above 50% on the Micro-GPS database, retrieving 75% of images with at least 20% overlap, and 97% of images with at least 60% overlap. Using those retrievals

for global localization without prior increases the success rates of StreetMap and our GTBL Method slightly on the Micro-GPS database and significantly on the larger HD Ground Database, compared to the case in which all reference images are considered.

In Chapter 6, we identified the number of extracted features per reference image as one of the most important parameters to optimize for feature-based localization methods. A larger number of stored reference features tends to improve the localization success rate, as it increases the chance of finding correct feature correspondences. However, it directly increases computation time and memory consumption of the localization methods. A proper selection of the number of stored features per reference image depends on the feature repeatability of the employed feature extraction and matching pipeline as defined in Section 8.2, which directly influences the inlier-to-outlier ratio among the the proposed correspondences, and it depends on the available compute power, time for localization, and memory.

Another trade-off is between the use of texture-specific parameters and generally suitable ones. Here, the choice depends mainly on the texture sensitivity, respectively the parametrization sensitivity, of the method, and the required effort for finding texture-specific parametrization. We observed good generalization capabilities of Ranger and StreetMap in Chapter 8, and also of our DML image retrieval approach in Chapter 7. The GTBL methods, on the other hand, presented significantly better performance on the textures they have been trained on, compared to the performance on the generalization textures of the HD Ground Database. In this case, however, our automatic parametrization framework, which we proposed in Chapter 6, can be used for efficient parametrization with only a few test images per texture.

What are the biggest challenges for an autonomous agent relying on ground texture based localization? From the technical side, high vehicle speeds are a challenge, because, as we derived in Section 3.3.1, it necessitates correspondingly short exposure times to avoid significant amounts of motion blur. For our mobile platform, for example, we derived a maximum exposure time of 0.09 ms to support vehicle speeds of up to 20 km/h. For a street car with a similar camera setup, we would require maximum exposure times of 0.015 ms, if, for example, we would like to support speeds of up to 120 km/h with similar image quality as that of our setup. Such short exposure times require a correspondingly bright illumination. If a cross-polarization solution is employed to avoid specular reflections, e. g. on a wet street, this requirement is further reinforced.

In Chapter 7, we observed a particularly large challenge in the localization task if the state of the surface changed between the recording of reference images and the recording of the query image. This change of state may be caused by weather, e. g. having a dry surface during mapping and wet surface at localization time, or it may be caused by wear-and-tear that occurs over time. We observed both of these challenges on the outdoor textures of the HD Ground Database deteriorating the localization success rates of all evaluated localization methods. We observed in Chapter 4 that the employed feature extractors are robust to the employed synthetic geometric and photometric image transformations. Also, we observed in Chapter 5 that the evaluated localization methods are robust to the natural changes of the ground between mapping and localization that occur in the Micro-GPS database of Zhang et al. [2019], i. e. varying camera positioning and orientation and small time intervals between the image recordings. But, it seems that the changes covered in the HD Ground Database, which appear outdoor after long time intervals or after drastic weather changes, are sufficient to shift the observed ground features to such an extent that finding correspondences becomes a major challenge. Here, a map update mechanism could help to deal with appearance changes occurring over time, i. e. the map is updated regularly based on the localized query images. A possible approach to deal with appearance changes occurring due to weather changes would be to design visual features robust to these changes or the employment of multiple maps, one for each relevant state of the ground, e. g. one map recorded with the ground being dry and one recorded with the ground being wet.

Another challenge for ground texture based localization is the initial global localization, where no prior pose estimate is available. Or in other words: the less accurate the prior, the more demanding becomes the localization task. We observed this in Chapter 5 for the wooden texture of the Micro-GPS database and on all application areas of the HD Ground Database, all of which are covered by many more reference images as those of the Micro-GPS database. This can be attributed to visual aliasing that becomes a challenge with increasing numbers of considered reference images. It means that the localization methods are getting confused with similar looking ground patches. The more reference images are considered, the more often this occurs, and the chance of confusing the actually observed ground area with another area increases. Whether this becomes a problem in practice depends mainly on the employed localization method, the ground texture of the application area, and the actual number of considered reference images. For the number of considered reference images, we examined three possible ways of reducing it, all of which allowed us to

improve localization performance significantly: (a) considering only those images in spatial proximity to a prior pose estimate (Chapter 5 and 8), (b) using image retrieval (Chapter 7), and (c) avoiding a complete coverage of the application area with a teach-and-repeat scenario in which the application area is reduced to a single path, (Chapter 5).

An additional advantage of the teach-and-repeat scenario is that it removes the need for a full coverage scanning of large application areas. The full coverage is challenging because the recorded area per image might be small for a downward-facing camera, which leads to a large number of images that have to be recorded in a structured manner, and because map construction becomes a compute heavy task for large application areas.

Building a large map of the ground is also challenging because it suffers from the aforementioned visual aliasing problem. This can be avoided if the images are recorded in a structured way that allows to derive accurate initial pose estimates, which means that the search for correspondences can be reduced to the local proximity of the images. Furthermore, if the map consists of a large number of images, a path between two points in the map may pass a correspondingly large number of image borders, even if it is the shortest possible path. On its way, this path accumulates small errors of the estimated relative poses of neighboring images. This is why a map created in an image stitching process may only guarantee local consistency. We can accumulate arbitrarily large distortions for sufficiently long paths on large maps if constructed as a simple image stitching. A possible approach to address this problem could be to include absolute reference points of the application area into the map. Another approach might be to consider the input of other drift-free sensors like GNSS, an inertial measurement unit, or a compass.

If the entire map is available during localization, having large maps with many reference images can consume a lot of memory. To avoid memory issues the agent could keep only a local map, i. e. a small part of the map from the current local vicinity, in the quickly available memory. Whenever the agent moves towards the border of the local map it is updated accordingly from a mass storage which may be on board or external to the robot. Alternatively, some parts or the entire localization capability may be performed externally.

9.2 Outlook

This section presents promising directions of future research in the area of ground texture based localization.

Development of a feature extraction pipeline for ground images In Chapter 4, we found several existing handcrafted feature extraction pipelines being well suited for ground texture based localization, also we observed very good performance of multiple localization methods based on those features in Chapter 5. However, Zhang and Rusinkiewicz [2018] and also we in Chapter 7 observed potential improvements processing the images with deep learning pipelines. There, we also observed significant difficulties of the existing localization methods in the task of initial localization on the HD Ground Database. In particular localization on the outdoor textures, if the ground state is altered over time or due to weather, is challenging for the existing methods. These difficulties arise from a failure in finding correct feature correspondences in actually overlapping images pairs that are subject to significant alterations. The promising results of our deep metric learning approach for image retrieval of ground images suggest that deep learning approaches, being specifically trained on ground images, present a possible solution for further improvements.

Map update mechanism Another potential approach to tackle the observed difficulties in the localization tasks with the state of the ground being altered significantly between scanning of the ground and subsequent localization would be a map management system with an integrated map update mechanism. Such a system could be applied in a scenario in which the ground is recorded on a regular basis, e. g. because robots are taking images for their localization capabilities. The system could then use these images to keep track of changes of the ground and a map update mechanism would allow to integrate them into the reference database, e. g. by removing old reference images while adding new ones, or by removing and adding individual reference features. For this purpose, the system could keep track of how often a reference feature takes part in a successful localization attempt and how often in unsuccessful ones. An assessment could be derived from these statistics to decide whether old reference features from the database are updated with newer ones; for example, we could exchange them with the query image features that participated in a successful localization attempt.

A low-cost application-specific implementation While it might be of interest to develop the most optimal recording setup for ground texture based localization, a low-cost variant may also be of interest in practice. In Chapter 5, we observed that the available resolution of the HD Ground Database was not required. Rather, on most textures, a significantly reduced resolution of only 20% (0.50 mm pixel length) to 40% (0.25 mm pixel length) of the native resolution achieved better localization performance. Also, in many cases it is not required to support vehicle speeds of up to 20 km/h. Accordingly, it would be of interest to develop a minimal system that is still sufficient for the localization task.

Moving the localization capability away from the robot In order to further reduce the requirements on the capabilities of the robot, that is supposed to be localized based on ground features, the localization capability could be moved away from the robot platform onto an external server. As suggested by Kim et al. [2019], due to possible interruptions and latencies in the communication with the server, it might still be useful to perform incremental localization updates on the robot itself. The map-based localization could however be performed on the server, which would free-up some computation power on the robot and it would remove the need to store the entire set of reference images, respectively the extracted reference image features, on the robot itself. Alternatively, the available set of reference images on the robot could be kept according to the region the robot is currently located in. This would allow to store the reference images on a slow mass storage on the robot or again on an external server.

A Appendix

Contents of this chapter were partially published in [Schmid et al., 2019], [Schmid et al., 2020a], [Schmid et al., 2020b], [Schmid et al., 2022], [Radhakrishnan et al., 2021], and [Schmid et al., 2021].

A.1 Local Visual Features for Ground Texture Based Localization: Parameter Settings

This section describes our parameter optimization strategy and the obtained optimized parameters for our survey of suitable features for ground texture based localization in Chapter 4.

In order to find the best parameter settings of the evaluated methods, we extract features from synthetically transformed images using varying parameter settings, and evaluate the obtained results based on a few selected key performance metrics.

A set of 3 non-overlapping images has been selected for each of the 6 examined texture types (fine asphalt, coarse asphalt, carpet, concrete, tiles, and wood). This results in a total number of 18 evaluated images, which are selected from a training set that is not included during testing. Each image is synthetically transformed as for testing (overall 72 synthetic transformations are tested), which results in 1296 pairs of reference and transformed images.

For keypoint detector parameter optimization, we introduce *adjusted repeatability* as a novel performance metric, which is a derived metric that combines the conventional repeatability metric of Mikolajczyk et al. [2005] and our ambiguity score, which was defined in Section 4.3.1:

$$\text{adjusted repeatability} = \frac{\text{repeatability}}{\text{ambiguity}}. \tag{A.1}$$

Table A.1: Optimized parameter settings for the evaluated **detection-only methods**, i. e. methods that extract keypoint objects but cannot be used for feature description. We employ the implementations of the OpenCV library [Bradski, 2000]; accordingly, the presented parameters are named as in the library. The reader may refer to the corresponding documentation for a detailed description of the parameter usage.

Keypoint detector	Parameters
CenSurE [Agrawal et al., 2008] Implemented in OpenCV as **StarDetector**	`maxSize`=11, `responseThreshold`=0, `lineThresholdProjected`=27, `lineThresholdBinarized`=24, `suppressNonmaxSize`=4
FAST [Rosten and Drummond, 2006]	`threshold`=5, `nonmaxSuppression`=true, `type`=TYPE_9_16
AGAST [Mair et al., 2010]	`threshold`=5, `nonmaxSuppression`=false, `type`=TYPE_7_12s
MSER [Matas et al., 2004]	`_delta`=0, `_min_area`=160, `_max_area`=14400, `_max_variation`=0.02
MSD [Tombari and Di Stefano, 2014]	`m_patch_radius`=3, `m_search_area_radius`=3, `m_nms_radius`=5, `m_nms_scale_radius`=0, `m_th_saliency`=30, `m_kNN`=50, `m_scale_factor`=4.5, `m_n_scales`=−1, `m_compute_orientation`=false
H.L. [Mikolajczyk and Schmid, 2002]	`numOctaves`=6, `corn_thresh`=0.0008, `DOG_thresh`=0.001, `num_layers`=2
GFTT [Shi and Tomasi, 1994]	`qualityLevel`=0.01, `minDistance`=5, `blockSize`=5, `useHarrisDetector`=true, `k`=0.01

The advantage of adjusted repeatability over conventional repeatability is that it does not reward clustering of keypoint objects. The following example illustrates the advantage of adjusted repeatability for parameter optimization. Repeatability increases if keypoint objects are assigned large associated regions, because it becomes more likely that a pair of keypoint objects from the reference image and the test image have an IoU greater 0.5. In the extreme case, if each keypoint object region is as large as the entire image, repeatability takes the *optimal* value of 1.0. This behavior of the repeatability metric is misleading during parameter optimization. Ambiguity, on the other hand, is also increasing for larger keypoint object regions. Therefore, adjusted repeatability takes a small value if each keypoint object is as large as the whole image, which makes it a more suited performance metric for detector parameter optimization.

We decide to optimize keypoint detector parameters for: 1. < 100 KPs, 2. adjusted repeatability, and 3. detection time. In order to obtain the best parameter

Table A.2: Optimized parameter settings for the evaluated **description-only methods**, i. e. methods that can be used for feature description but not to detect keypoint objects. We employ the implementations of the OpenCV library [Bradski, 2000]; accordingly, the presented parameters are named as in the library. The reader may refer to the corresponding documentation for a detailed description of the parameter usage.

Feature description method	Parameters
DAISY [Tola et al., 2010]	`radius`=5, `q_radius`=3, `q_theta`=8, `q_hist`=10, `norm`=NRM_NONE, `interpolation`=true, `use_orientation`=true
BRIEF [Calonder et al., 2010]	`bytes`=32, `use_orientation`=true
FREAK [Alahi et al., 2012]	`orientationNormalized`=true, `scaleNormalized`=false, `patternScale`=17, `nOctaves`=2
LATCH [Levi and Hassner, 2016]	`bytes`=64, `rotationInvariance`=true, `half_ssd_size`=6, `sigma`=3.6

settings, we adapt the parameters manually for each detector until performance peaks. Table A.1 presents the final parameter settings of the methods taken from OpenCV [Bradski, 2000] and the evaluated ORB implementation from ORB-SLAM2 [Mur-Artal and Tardós, 2017]. For methods that allow to specify the number of keypoint objects to retrieve, we put this value to 1000; otherwise, NMS was used to reduce the number of keypoint objects to 1000.

To optimize the parameters of feature description methods, we employ SURF [Bay et al., 2006] to provide the keypoint objects (besides for feature description with AKAZE, where it had to be the AKAZE detection method) and evaluate pose estimation performance on synthetically transformed images. The parameters shown in Table A.2 are optimized for 1. pose estimation success rate, 2. matching precision, 3. number of correct feature matches.

Parameters for feature extractors that can perform keypoint detection as well as feature description are presented in Table A.3. We optimized these methods first for keypoint detection and then for feature description, using the same optimization strategies previously described.

Table A.3: Optimized parameter settings for the evaluated **feature extractors**, i. e. methods that can be used to detect keypoint objects and also to describe features. For ORB, we use the implementation from ORB-SLAM2 [Mur-Artal and Tardós, 2017]. Otherwise, we employ the OpenCV library [Bradski, 2000]. The reader may refer to the corresponding documentations for further details about the parameters.

Feature extractor	Parameters
SIFT [Lowe, 2004]	`nOctaveLayers`=12, `contrastThreshold`=0.003, `edgeThreshold`=9, `sigma`=8.7
SURF [Bay et al., 2006]	`hessianThreshold`=20, `nOctaves`=1, `nOctaveLayers`=2, `extended`=false, `upright`=false
ORB [Rublee et al., 2011, Mur-Artal and Tardós, 2017]	`scaleFactor`=1.0, `nlevels`=1, `iniThFAST`=29, `minThFAST`=3
BRISK [Leutenegger et al., 2011]	`thresh`=6, `octaves`=1, `patternScale`=1.45
AKAZE [Alcantarilla et al., 2013]	`descriptor_type`=DESCRIPTOR_MLDB, `descriptor_size`=486, `descriptor_channels`=3, `threshold`=0.0001, `nOctaves`=2, `nOctaveLayers`=2, `diffusivity`=DIFF_CHARBONNIER

A.2 Identity Matching with Compact Binary Descriptors: Parameter Settings

This section presents the optimized parameters that we obtained for the GTBL method, StreetMap, and Ranger, for the evaluation on the HD Ground Database in Section 5.2.2 and Section 8.4.2.

A.2.1 GTBL Method

For the GTBL method, we optimize the employed image scale, the number of extracted query and reference image features, the histogram grid cell size of the voting procedure, the number of LATCH [Levi and Hassner, 2016] bits considered for identity matching, and the SIFT [Lowe, 2004] parameters of the employed OpenCV 4.0 [Bradski, 2000] implementation `nOctaveLayers`, `contrastThreshold`, `edgeThreshold`, and `sigma`, as well as the corresponding OpenCV 4.0 LATCH parameters `half_ssd_size`, and `sigma`.

The values, that we obtain with texture-specific optimization and with a generalized optimization that jointly considers all training areas of the HD Ground

Table A.4: The employed GTBL method [Schmid et al., 2020a] (later called GTBL SIFT) parameter settings on the HD Ground Database [Schmid et al., 2022]. The grid cell size is specified in millimeters for its corresponding size in the real world, and in pixels for its size in the image at the texture-specific applied image scale.

	Asphalt	Cobblestone	Carpet	Laminate	Generalized
Scale	0.60	0.34	0.35	0.20	0.24
#Q. image features	600	600	600	1100	950
#Ref. image features	850	750	600	1400	850
Grid cell size (pixels)	115	210	150	300	310
Grid cell size (mm)	≈ 19.2	≈ 61.8	≈ 42.9	≈ 150.0	≈ 129.2
LATCH bits	21	20	20	20	19
`nOctaveLayers`	9	21	13	15	7
`contrastThreshold`	0.045	0.02	0.02	0.023	0.015
`edgeThreshold`	18	18	13	14	17
`sigma` (SIFT)	4.5	2.8	5.4	2.0	2.6
`half_ssd_size`	4	3	6	3	4.0
`sigma` (LATCH)	3.4	3.4	4.0	4.8	4.0

Database, are shown in Table A.4.

A.2.2 StreetMap

For StreetMap, we optimize the employed image scale, the number of extracted image features, the factor applied by the ratio test for outlier rejection, and the OpenCV 4.0 SURF [Bay et al., 2006] parameters: `hessianThreshold`, `nOctaves`, and `nOctaveLayers`. Results for the case of texture-specific and generalized optimization are presented in Table A.5.

A.2.3 Ranger

One parameter for Ranger is the number of required feature matches between the query image and the currently considered reference image that have to remain after the cross-check in order to estimate the query image pose based on these matches. The default value for this parameter suggested by the authors is 25, which is also the value we used for the evaluation on the Micro-GPS database. For the HD Ground Database, the value set as follows after optimization: 10 for asphalt, 20 for cobblestone, 8 for carpet, 10 for laminate, and 23 in the generalized case.

Furthermore, we optimize the scale, the number of extracted features per image, and the parameters of the OpenCV 4.0 CenSurE [Agrawal et al., 2008] imple-

Table A.5: The employed StreetMap [Chen et al., 2018] parameter settings on the HD Ground Database [Schmid et al., 2022].

	Asphalt	Cobblestone	Carpet	Laminate	Generalized
Scale	0.20	0.38	0.20	0.70	0.30
#Image features	200	400	600	900	1150
Ratio Test Factor	0.825	0.85	0.89	0.875	0.89
`hessianThreshold`	36	55	14	47	70
`nOctaves`	1	1	5	9	1
`nOctaveLayers`	2	1	3	2	3

mentation called StarDetector. The texture-specific and generalized parameter settings are shown in Table A.6.

A.3 Model-Based Parameter Optimization: Derivation of the Prediction Model

We model the probability of successful localization for methods that have the properties described in Section 6.2 This derivation was published in [Schmid et al., 2021].

Let the random variable $N_{\mathcal{M}}$ denote the number of proposed feature matches, where $N_{\mathcal{M}} = N_{\mathcal{I}} + N_{\mathcal{O}}$, with $N_{\mathcal{I}}$ and $N_{\mathcal{O}}$ denoting the numbers of inliers, respectively outliers. Furthermore, let $\mathcal{V}$ denote the set of voting grid cells that received at least one vote. For a voting cell $v \in \mathcal{V}$, $N^v_{\mathcal{M}}$ denotes the random variable that represents the number of votes cast onto it, i. e. the number of matches with a corresponding pose estimate that projects the query image position into the boundaries of voting cell v. Similarly, $N^v_{\mathcal{I}}$ and $N^v_{\mathcal{O}}$ represent respectively the number of inliers and outliers among them. The voting peak is

Table A.6: The employed Ranger [Kozak and Alban, 2016] parameter settings on the HD Ground Database [Schmid et al., 2022].

	Asphalt	Cobblestone	Carpet	Laminate	Generalized
Scale	0.20	0.88	0.20	0.28	0.60
#Image features	400	650	350	350	1000
`maxSize`	5	8	6	6	13
`responseThreshold`	14	15	9	9	9
`lineThresholdProjected`	21	22	30	30	45
`lineThresholdBinarized`	34	22	16	16	55
`suppressNonmaxSize`	12	9	8	8	31

denoted as

$$v_p = \{v \in \mathcal{V} | N_{\mathcal{M}}^{v} = \max_{v' \in \mathcal{V}} N_{\mathcal{M}}^{v'}\}, \tag{A.2}$$

assuming there is a single voting cell with most votes.

If we assume that the pose estimation algorithm works well, localization succeeds when having two or more inliers on the voting peak, as two correct matches are sufficient to determine the correct query image pose. Generally, a single inlier is insufficient as it might be indistinguishable from an outlier. Therefore, we model the localization success rate as

$$\Pr[N_{\mathcal{I}}^{v_p} \geq 2]. \tag{A.3}$$

Let $\mathcal{V}_{\mathcal{I}} \subset \mathcal{V}$ denote the subset of voting cells with at least one inlier vote. If one of those voting cells receives two or more inlier votes and has overall more votes than any other voting cell, localization succeeds. For a voting cell $v_i \in \mathcal{V}_{\mathcal{I}}$, we compute the probability of this condition being true as

$$\Pr[(N_{\mathcal{I}}^{v_i} \geq 2) \cap (N_{\mathcal{M}}^{v_i} = N_{\mathcal{M}}^{v_p})], \tag{A.4}$$

where $\Pr[\mathcal{A} \cap \mathcal{B}]$ denotes the probability of both events $\mathcal{A}$ and $\mathcal{B}$ to occur together. Localization succeeds if it is *not* the case that this condition is *not* true for any $v_i \in \mathcal{V}_{\mathcal{I}}$:

$$\begin{aligned} &\Pr[N_{\mathcal{I}}^{v_p} \geq 2] = \\ &1 - \left(\prod_{v_i \in \mathcal{V}_{\mathcal{I}}} \left[1 - \left(\Pr[(N_{\mathcal{I}}^{v_i} \geq 2) \cap (N_{\mathcal{M}}^{v_i} = N_{\mathcal{M}}^{v_p})] \right) \right] \right). \end{aligned} \tag{A.5}$$

In order to compute the probability of having j votes on voting cell $v \in \mathcal{V}$, we consider the number of inliers $N_{\mathcal{I}}^{v}$ and the number of outliers $N_{\mathcal{O}}^{v}$ on it:

$$\Pr[N_{\mathcal{M}}^{v} = j] = \sum_{k=0}^{j} \left[\Pr[N_{\mathcal{I}}^{v} = k] \cdot \Pr[N_{\mathcal{O}}^{v} = j - k | N_{\mathcal{I}}^{v} = k] \right], \tag{A.6}$$

with $\Pr[\mathcal{A}|\mathcal{B}]$ denoting the conditional probability of $\mathcal{A}$ given $\mathcal{B}$. As a next step, we make an assumption about the distribution of outlier votes. Here, we assume to have Complete Spatial Randomness (CSR) among the voting positions, i. e. the probability $p_{\text{out\_vote}}$ of any outlier match $m \in \mathcal{O}$, casting a vote on the voting cell v is the same for any voting cell $v \in \mathcal{V}$. Furthermore, we assume to have many matches. So, we can assume statistical independence for any two voting

cells $v_1, v_2 \in \mathcal{V}$

$$\Pr[N_{\mathcal{M}}^{v_1} = i | N_{\mathcal{M}}^{v_2} = j] = \Pr[N_{\mathcal{M}}^{v_1} = i]\,. \tag{A.7}$$

Also, assuming to have significantly more outliers than inliers, we approximate

$$\Pr[N_{\mathcal{O}}^{v} = i | N_{\mathcal{I}}^{v} = j] = \Pr[N_{\mathcal{O}}^{v} = i]\,. \tag{A.8}$$

Based on the CSR assumption, $N_{\mathcal{O}}^{v}$ for $v \in \mathcal{V}$ is binomially distributed. The distribution is characterized by the number of outliers $N_{\mathcal{O}}$, and the probability $p_{\text{out\_vote}} = 1/|\mathcal{V}|$ that each of them has for casting a vote on v:

$$\Pr[N_{\mathcal{O}}^{v} = i] = \mathrm{B}(i | p_{\text{out\_vote}}, N_{\mathcal{O}})\,, \tag{A.9}$$

where $\mathrm{B}(i|\mathrm{p}, n)$ denotes the probability of observing i successes in n independent Bernoulli trials, each with a success probability of p.

To estimate $N_{\mathcal{I}}^{v}$, the number of inliers casting a vote on $v \in \mathcal{V}$, we assume that every extracted query feature $f_q \in \mathcal{F}_{\mathrm{q}}$ has the same probability $p_{\text{in\_vote}}^{v}$ of generating one inlier vote on v (with $p_{\text{in\_vote}}^{v} = 0$ for any $v \in \mathcal{V} \setminus \mathcal{V}_{\mathcal{I}}$), and we assume that no query feature will generate more than one inlier vote for v. Accordingly, the random variable $N_{\mathcal{I}}^{v}$ is binomially distributed as well, depending on $p_{\text{in\_vote}}^{v}$ and the number of extracted query image features $|\mathcal{F}_{\mathrm{q}}|$:

$$\Pr[N_{\mathcal{I}}^{v} = i] = \mathrm{B}(i | p_{\text{in\_vote}}^{v}, |\mathcal{F}_{\mathrm{q}}|)\,. \tag{A.10}$$

The upper limit for the number of votes any voting cell can receive is $N_{\mathcal{M}}$. Considering this and Eq. (A.7), we estimate

$$\Pr[N_{\mathcal{M}}^{v_i} = N_{\mathcal{M}}^{v_p}] = \sum_{j=1}^{N_{\mathcal{M}}} \left[\Pr[N_{\mathcal{M}}^{v_i} = j] \cdot \prod_{v \in \mathcal{V} \setminus \{v_i\}} \Pr[N_{\mathcal{M}}^{v} < j] \right]\,. \tag{A.11}$$

Using Eq. (A.6) and Eq. (A.8), we obtain

$$\begin{aligned} &\Pr[N_{\mathcal{M}}^{v_i} = N_{\mathcal{M}}^{v_p}] = \\ &\sum_{j=1}^{N_{\mathcal{M}}} \left[\sum_{k=0}^{j} \left[\Pr[N_{\mathcal{I}}^{v_i} = k] \cdot \Pr[N_{\mathcal{O}}^{v_i} = j-k] \right] \cdot \prod_{v \in \mathcal{V} \setminus \{v_i\}} \Pr[N_{\mathcal{M}}^{v} < j] \right]\,. \end{aligned} \tag{A.12}$$

Now, we can use this in Eq. (A.4) to obtain

$$\begin{aligned}&\Pr[(N_{\mathcal{I}}^{v_i} \geq 2) \cap (N_{\mathcal{M}}^{v_i} = N_{\mathcal{M}}^{v_p})] = \\ &\sum_{j=2}^{N_{\mathcal{M}}} \left[\sum_{k=2}^{j} \left[\Pr[N_{\mathcal{I}}^{v_i} = k] \cdot \Pr[N_{\mathcal{O}}^{v_i} = j-k] \right] \cdot \prod_{v \in \mathcal{V} \setminus \{v_i\}} \Pr[N_{\mathcal{M}}^{v} < j] \right].\end{aligned} \tag{A.13}$$

Finally, considering Eq. (A.13) with the substitution

$$\begin{aligned}\Pr[N_{\mathcal{M}}^{v} < j] &= \sum_{k=0}^{j-1} \Pr[N_{\mathcal{M}}^{v} = k] \\ &= \sum_{k=0}^{j-1} \left[\sum_{l=0}^{k} \left[\Pr[N_{\mathcal{I}}^{v} = l] \cdot \Pr[N_{\mathcal{O}}^{v} = k-l] \right] \right],\end{aligned} \tag{A.14}$$

we model the probability of observing a successful localization attempt using Eq. (A.5) as:

$$\Pr[N_{\mathcal{I}}^{v_p} \geq 2] = 1 - x, \tag{A.15}$$

with

$$\begin{aligned}x &= \prod_{v_i \in \mathcal{V}_{\mathcal{I}}} \left[1 - \left(\Pr[(N_{\mathcal{I}}^{v_i} \geq 2) \cap (N_{\mathcal{M}}^{v_i} = N_{\mathcal{M}}^{v_p})] \right) \right] \\ &= \prod_{v_i \in \mathcal{V}_{\mathcal{I}}} \left[1 - \left(\sum_{j=2}^{N_{\mathcal{M}}} \left[\sum_{k=2}^{j} \left[\Pr[N_{\mathcal{I}}^{v_i} = k] \cdot \Pr[N_{\mathcal{O}}^{v_i} = j-k] \right] \cdot \right. \right. \right. \\ &\quad \left. \left. \left. \prod_{v \in \mathcal{V} \setminus \{v_i\}} \sum_{k=0}^{j-1} \left[\sum_{l=0}^{k} \left[\Pr[N_{\mathcal{I}}^{v} = l] \cdot \Pr[N_{\mathcal{O}}^{v} = k-l] \right] \right] \right) \right] \right].\end{aligned} \tag{A.16}$$

A.4 Deep Metric Learning for Global Localization: BoW Parameter Settings

For our Bag of Words (BoW) [Galvez-López and Tardos, 2012] implementation presented in Section 7.3.2.2 using SIFT features [Lowe, 2004], we show the results of our parameter optimization for the texture-specific optimal choice of n, i. e. the number of features that are extracted per image.

Table A.7 presents the observed recall performance on the Micro-GPS database [Zhang et al., 2019]. It also presents the resulting texture-specific choice for n.

Table A.7: Evaluation of the R@100 performance, i. e. the recall when retrieving 100 reference images per query image, of our Bag of Words (BoW) [Galvez-López and Tardos, 2012] implementation on the Micro-GPS database [Zhang et al., 2019]. For varying numbers of extracted features per image, we present the texture-specific results. The values highlighted in bold are the ones that are used for the respective textures when we apply the BoW procedure for image retrieval.

Texture	Number of features – R@100(%)									
	100	200	300	400	500	600	700	800	900	1000
Carpet	20.5	22.9	19.1	19.8	22.6	23.5	18.5	**29.0**	17.9	17.2
Coarse Asphalt	27.1	24.6	27.4	20.4	22.6	**28.8**	17.4	24.1	24.3	24.5
Concrete	12.2	7.8	17.1	12.9	8.7	20.1	5.7	**20.9**	10.7	13.5
Fine Asphalt	**25.7**	19.9	22.5	12.4	20.5	18.7	11.9	15.6	23.5	14.2
Tiles	8.9	18.1	17.7	12.8	11.1	14.1	17.0	**20.5**	16.6	16.0
Wood	6.1	9.4	9.2	**11.4**	5.7	11.3	8.6	8.3	7.9	9.4

Table A.8: The employed GTBL RND [Schmid et al., 2020b] parameter settings on the HD Ground Database [Schmid et al., 2022]. The grid cell size is specified in millimeters for its corresponding size in the real world, and in pixels for its size in the image at the texture-specific applied image scale.

	Asphalt	Cobblestone	Carpet	Laminate	Generalized
Scale	0.20	0.23	0.23	0.15	0.15
#Q. image features	3500	2200	2100	2300	3300
#Ref. image features	5400	4400	2400	3800	4800
Grid cell size (pixels)	250	240	210	295	190
Grid cell size (mm)	$\approx$ 125.0	$\approx$ 104.3	$\approx$ 91.3	$\approx$ 196.7	$\approx$ 126.7
LATCH bits	20	18	17	19	20
`half_ssd_size`	7	5	9	2	4
`sigma` (LATCH)	3.8	3.0	2.2	5.0	2.9

A.5 Faster Local Localization with Keypoint Sampling: Parameter Settings

Here, we present the optimized parameters that we use for the GTBL RND method for the evaluation on the HD Ground Database in Section 8.4.2.

We optimize the same values as for the GTBL Method, whereby in this case the SIFT parameter setting does not matter. Our texture-specific settings, as well as the settings for the generalized case, are shown in Table A.8.

Bibliography

Agarwal, S., Mierle, K., and Others (2016). Ceres solver. http://ceres-solver.org.

Agrawal, M., Konolige, K., and Blas, M. R. (2008). CenSurE: Center surround extremas for realtime feature detection and matching. In *IEEE European Conference on Computer Vision (ECCV)*, pages 102–115, Berlin, Heidelberg. Springer Berlin Heidelberg.

Alahi, A., Ortiz, R., and Vandergheynst, P. (2012). FREAK: Fast retina keypoint. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 510–517.

Alcantarilla, P. F., Bartoli, A., and Davison, A. J. (2012). KAZE features. In *IEEE European Conference on Computer Vision (ECCV)*, pages 214–227, Berlin, Heidelberg. Springer Berlin Heidelberg.

Alcantarilla, P. F., Nuevo, J., and Bartoli, A. (2013). Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *Proceedings of the British Machine Vision Conference (BMVC)*.

Aqel, M. O. A., Marhaban, M. H., Saripan, M. I., Ismail, N. B., and Khmag, A. (2016). Optimal configuration of a downward-facing monocular camera for visual odometry. *Indian Journal of Science and Technology*, 8(32).

Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., and Sivic, J. (2016). NetVLAD: CNN architecture for weakly supervised place recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Avrithis, Y. and Tolias, G. (2014). Hough pyramid matching: Speeded-up geometry re-ranking for large scale image retrieval. *International Journal of Computer Vision (IJCV)*, 107(1):1–19.

Babenko, A., Slesarev, A., Chigorin, A., and Lempitsky, V. (2014). Neural codes for image retrieval. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *IEEE European Conference on Computer Vision (ECCV)*, pages 584–599.

Bailo, O., Rameau, F., Joo, K., Park, J., Bogdan, O., and Kweon, I. S. (2018). Efficient adaptive non-maximal suppression algorithms for homogeneous spatial keypoint distribution. *Pattern Recognition Letters*, 106:53 – 60.

Balntas, V., Li, S., and Prisacariu, V. (2018). RelocNet: Continuous metric learning relocalisation using neural nets. In *IEEE European Conference on Computer Vision (ECCV)*.

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). SURF: Speeded up robust features. In *IEEE European Conference on Computer Vision (ECCV)*, pages 404–417, Berlin, Heidelberg. Springer Berlin Heidelberg.

Blanco, J. L. and Rai, P. K. (2014). nanoflann: a C++ header-only fork of FLANN, a library for nearest neighbor (NN) with KD-trees. https://github.com/jlblancoc/nanoflann.

Bradski, G. (2000). The OpenCV library. *Dr. Dobb's Journal of Software Tools*.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. (1994). Signature verification using a Siamese time delay neural network. In *Advances in Neural Information Processing Systems*, volume 6.

Calonder, M., Lepetit, V., Strecha, C., and Fua, P. (2010). BRIEF: Binary robust independent elementary features. In *IEEE European Conference on Computer Vision (ECCV)*, pages 778–792, Berlin, Heidelberg. Springer Berlin Heidelberg.

Cao, B., Araujo, A., and Sim, J. (2020). Unifying deep local and global features for image search. In *IEEE European Conference on Computer Vision (ECCV)*, pages 726–743.

Chatoux, H., Lecellier, F., and Fernandez-Maloigne, C. (2016). Comparative study of descriptors with dense key points. In *International Conference on Pattern Recognition (ICPR)*, pages 1988–1993.

Chen, P., Gu, Z., Zhang, G., and Liu, H. (2014). Ceiling vision localization with feature pairs for home service robots. In *IEEE Transactions on Circuits and Systems for Video Technology (ROBIO)*, pages 2274–2279.

Chen, X., Vempati, A. S., and Beardsley, P. (2018). StreetMap - mapping and localization on ground planes using a downward facing camera. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1672–1679.

Chopra, S., Hadsell, R., and LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 539–546.

Cornick, M., Koechling, J., Stanley, B., and Zhang, B. (2016). Localizing ground penetrating RADAR: A step toward robust autonomous ground vehicle localization. *Journal of Field Robotics*, 33(1):82–102.

Desouza, G. N. and Kak, A. C. (2002). Vision for mobile robot navigation: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 24(2):237–267.

Duan, L., Lin, J., Wang, Z., Huang, T., and Gao, W. (2015). Weighted component hashing of binary aggregated descriptors for fast visual search. *IEEE Transactions on Multimedia*, 17(6):828–842.

Fang, H., Yang, M., and Yang, R. (2007). Ground texture matching based global localization for intelligent vehicles in urban environment. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 105–110.

Fang, H., Yang, M., Yang, R., and Wang, C. (2009). Ground-texture-based localization for intelligent vehicles. *IEEE Transactions on Intelligent Transportation Systems (ITS)*, 10(3):463–468.

Fischler, M. A. and Bolles, R. C. (1987). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. In *Readings in Computer Vision*, pages 726 – 740. Morgan Kaufmann, San Francisco (CA).

Galvez-López, D. and Tardos, J. D. (2012). Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197.

Gilles, M. and Ibrahimpasic, S. (2021). Unsupervised deep learning based ego motion estimation with a downward facing camera. *The Visual Computer*, pages 1–14.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

Gordo, A., Almazan, J., Revaud, J., and Larlus, D. (2017). End-to-end learning of deep visual representations for image retrieval. *International Journal of Computer Vision*, 124(2):237–254.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Alvey Vision Conference (AVC)*, volume 15, pages 147–151.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hoffer, E. and Ailon, N. (2015). Deep metric learning using triplet network. In *Similarity-Based Pattern Recognition*, pages 84–92.

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jégou, H., Douze, M., Schmid, C., and Pérez, P. (2010). Aggregating local descriptors into a compact image representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3304–3311.

Kaya, M. and Bilge, H. S. (2019). Deep metric learning: A survey. *Symmetry*, 11(9).

Kelly, A. (2000). Mobile robot localization from large-scale appearance mosaics. *The International Journal of Robotics Research*, 19(11):1104–1125.

Kelly, A., Nagy, B., Stager, D., and Unnikrishnan, R. (2007). Field and service applications - an infrastructure-free automated guided vehicle based on computer vision - an effort to make an industrial robot vehicle that can operate without supporting infrastructure. *IEEE Robotics and Automation Magazine (RAM)*, 14(3):24–34.

Kendall, A. and Cipolla, R. (2017). Geometric loss functions for camera pose regression with deep learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6555–6564.

Kim, J., Kim, Y., Zewge, N. S., and Kim, J.-H. (2019). A robust client-server architecture for map information processing and transmission for distributed visual SLAM. In *International Conference on Robot Intelligence Technology and Applications (RiTA)*, pages 99–105.

Kitt, B., Geiger, A., and Lategahn, H. (2010). Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 486–492.

Kozak, K. C. and Alban, M. (2016). Ranger: A ground-facing camera-based localization system for ground vehicles. In *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 170–178.

Leutenegger, S., Chli, M., and Siegwart, R. Y. (2011). BRISK: Binary robust invariant scalable keypoints. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2548–2555.

Levi, G. and Hassner, T. (2016). LATCH: Learned arrangements of three patch codes. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9.

Liu, C., Yuen, J., Torralba, A., Sivic, J., and Freeman, W. T. (2008). SIFT Flow: Dense correspondence across different scenes. In *IEEE European Conference on Computer Vision (ECCV)*, pages 28–42, Berlin, Heidelberg. Springer Berlin Heidelberg.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110.

Macias-Sola, J., Uttendorf, S., and Blech, J. O. (2021). A ground texture-based mapping and localization method for AGVs. In *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–6.

Mair, E., Hager, G. D., Burschka, D., Suppa, M., and Hirzinger, G. (2010). Adaptive and generic corner detection based on the accelerated segment test. In *IEEE European Conference on Computer Vision (ECCV)*, pages 183–196, Berlin, Heidelberg. Springer Berlin Heidelberg.

Maree, R., Geurts, P., Piater, J., and Wehenkel, L. (2005). Random subwindows for robust image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 34–40 vol. 1.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761 – 767.

Mikolajczyk, K. and Schmid, C. (2002). An affine invariant interest point detector. In *IEEE European Conference on Computer Vision (ECCV)*, pages 128–142, Berlin, Heidelberg. Springer Berlin Heidelberg.

Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., and Gool, L. V. (2005). A comparison of affine region detectors. *International Journal of Computer Vision (IJCV)*, 65(1):43–72.

Mishchuk, A., Mishkin, D., Radenovic, F., and Matas, J. (2017). Working hard to know your neighbor's margins: Local descriptor learning loss. In *Advances in Neural Information Processing Systems*, pages 4826–4837.

Mount, J., Dawes, L., and Milford, M. (2019). Automatic coverage selection for surface-based visual localization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application (VISSAPP)*, pages 331–340. INSTICC Press.

Muñoz-Salinas, R. and Medina-Carnicer, R. (2020). UcoSLAM: simultaneous localization and mapping by fusion of keypoints and squared planar markers. *Pattern Recognition*.

Mur-Artal, R. and Tardós, J. D. (2017). ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262.

Nagai, I. (2007). Mobile robot with floor tracking device for localization and control. *Journal of Mechatronics and Robotics (JMR)*, 19:34–41.

Nagai, I. and Watanabe, K. (2015). Path tracking by a mobile robot equipped with only a downward facing camera. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6053–6058.

Nakashima, S., Morio, T., and Mu, S. (2019). AKAZE-based visual odometry from floor images supported by acceleration models. *IEEE Access*, 7:31103–31109.

Noh, H., Araujo, A., Sim, J., Weyand, T., and Han, B. (2017a). Large-scale image retrieval with attentive deep local features. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3476–3485.

Noh, H., Araujo, A., Sim, J., Weyand, T., and Han, B. (2017b). Large-scale image retrieval with attentive deep local features. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3476–3485.

Nourani-Vatani, N., Roberts, J., and Srinivasan, M. V. (2009). Practical visual odometry for car-like vehicles. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3551–3557.

Nowak, E., Jurie, F., and Triggs, B. (2006). Sampling strategies for bag-of-features image classification. In *IEEE European Conference on Computer Vision (ECCV)*, pages 490–503, Berlin, Heidelberg. Springer Berlin Heidelberg.

Otsu, K., Otsuki, M., Ishigami, G., and Kubota, T. (2013). *An Examination of Feature Detection for Real-Time Visual Odometry in Untextured Natural Terrain*, pages 405–414. Springer Berlin Heidelberg.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

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.

Perronnin, F. and Dance, C. (2007). Fisher kernels on visual vocabularies for image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

Pietikäinen, M., Hadid, A., Zhao, G., and Ahonen, T. (2011). *Computer vision using local binary patterns*, volume 40. Springer Science & Business Media.

Pion, N., Humenberger, M., Csurka, G., Cabon, Y., and Sattler, T. (2020). Benchmarking image retrieval for visual localization. In *International Conference on 3D Vision (3DV)*, pages 483–494.

Pratt, W. K. (2007). *Digital image processing: PIKS Scientific inside*, volume 4. Wiley-interscience Hoboken, New Jersey.

Radenović, F., Tolias, G., and Chum, O. (2019). Fine-tuning CNN image retrieval with no human annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 41(7):1655–1668.

Radhakrishnan, R., Schmid, J. F., Scholz, R., and Schmidt-Thieme, L. (2021). Deep metric learning for ground images. *arXiv preprint arXiv:2109.01569*.

Revaud, J., Almazan, J., Rezende, R., and Souza, C. (2019). Learning with average precision: Training image retrieval with a listwise loss. In *IEEE International Conference on Computer Vision (ICCV)*, pages 5106–5115.

Rodriguez, J. and Castano-Cano, D. (2019). SimSLAM 2D: A simulation framework for testing and benchmarking of two-dimensional visual-SLAM methods. In *International Conference on Advanced Robotics (ICAR)*, pages 141–147.

Rosten, E. and Drummond, T. (2006). Machine learning for high-speed corner detection. In *IEEE European Conference on Computer Vision (ECCV)*, pages 430–443, Berlin, Heidelberg. Springer Berlin Heidelberg.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2564–2571.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

Sánchez-Belenguer, C., Wolfart, E., and Sequeira, V. (2020). Rise: A novel indoor visual place recogniser. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 265–271.

Schmid, J. F., Simon, S. F., and Mester, R. (2019). Features for ground texture based localization – a survey. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 184.1–184.13.

Schmid, J. F., Simon, S. F., and Mester, R. (2020a). Ground texture based localization using compact binary descriptors. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1315–1321.

Schmid, J. F., Simon, S. F., and Mester, R. (2020b). Ground texture based localization: Do we need to detect keypoints? In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4575–4580.

Schmid, J. F., Simon, S. F., and Mester, R. (2021). Model-based parameter optimization for ground texture based localization methods. *arXiv preprint arXiv:2109.01559*.

Schmid, J. F., Simon, S. F., Radhakrishnan, R., Frintrop, S., and Mester, R. (2022). HD Ground - a database for ground texture based localization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 7628–7634.

Schönberger, J. L., Price, T., Sattler, T., Frahm, J.-M., and Pollefeys, M. (2017). A vote-and-verify strategy for fast spatial verification in image retrieval. In Lai, S.-H., Lepetit, V., Nishino, K., and Sato, Y., editors, *Asian Conference on Computer Vision (ACCV)*, pages 321–337, Cham. Springer International Publishing.

Shi, J. and Tomasi, C. (1994). Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593 – 600.

Smeulders, A., Worring, M., Santini, S., Gupta, A., and Jain, R. (2000). Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(12):1349–1380.

Soille, P. and Vincent, L. (1991). Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 13:583–598.

Surmann, H., Bredenfeld, A., Christaller, T., Frings, R., Petersen, U., and Wisspeintner, T. (2008). The volksbot. In *Workshop Proceedings of SIMPAR*, pages 551–561.

Swank, A. J. (2012). Localization using visual odometry and a single downward-pointing camera. *NASA*.

Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.

Thrun, S., Burgard, W., Fox, D., and Arkin, R. (2005). *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents series. MIT Press.

Tola, E., Lepetit, V., and Fua, P. (2010). DAISY: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 32(5):815–830.

Tolias, G., Sicre, R., and Jégou, H. (2016). Particular object retrieval with integral max-pooling of cnn activations.

Tombari, F. and Di Stefano, L. (2014). Interest points via maximal self-dissimilarities. In *Asian Conference on Computer Vision (ACCV)*, pages 586–600, Cham. Springer International Publishing.

Tuytelaars, T. (2010). Dense interest points. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2281–2288.

Tuytelaars, T. and Mikolajczyk, K. (2008). Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280.

van de Sande, K., Gevers, T., and Snoek, C. (2010). Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 32(9):1582–1596.

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., and Wu, Y. (2014). Learning fine-grained image similarity with deep ranking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1386–1393.

Xue, J., Zhang, H., Dana, K., and Nishino, K. (2017). Differential angular imaging for material recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yan, C. C., Xie, H., Zhang, B., Ma, Y., Dai, Q., and Liu, Y. (2015). Fast approximate matching of binary codes with distinctive bits. *Frontiers of Computer Science*, 9(5):741–750.

Yi, K. M., Trulls, E., Lepetit, V., and Fua, P. (2016). LIFT: Learned invariant feature transform. In *IEEE European Conference on Computer Vision (ECCV)*, pages 467–483.

Zaman, M. (2007). High precision relative localization using a single camera. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3908–3914.

Zeisl, B., Sattler, T., and Pollefeys, M. (2015). Camera pose voting for large-scale image-based localization. In *IEEE International Conference on Computer Vision (ICCV)*.

Zhang, L., Finkelstein, A., and Rusinkiewicz, S. (2017). High-precision localization using ground texture. *CoRR*, abs/1710.10687.

Zhang, L., Finkelstein, A., and Rusinkiewicz, S. (2019). High-precision localization using ground texture. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6381–6387.

Zhang, L. and Rusinkiewicz, S. (2018). Learning to detect features in texture images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.