Tampere University

JUNSHENG FU

Camera Pose Estimation from Street-view Snapshots and Point Clouds

Tampere University Dissertations 611

Tampere University Dissertations 611

JUNSHENG FU

Camera Pose Estimation from Street-view Snapshots and Point Clouds

ACADEMIC DISSERTATION
To be presented, with the permission of
the Faculty of Information Technology and Communication Sciences
of Tampere University,
for public discussion
on 3rd June 2022, at 12 o’clock.

ACADEMIC DISSERTATION
Tampere University, Faculty of Information Technology and Communication Sciences
Finland

Responsible supervisor and Custos	Professor Joni Kämäräinen Tampere University Finland	
Supervisor	Associate Professor Said Pertuz Universidad Industrial de Santander Colombia	
Pre-examiners	Professor Domenec Puig Universitat Rovira i Virgili Spain	Associate Professor Vedrana Dahl Technical University of Denmark Denmark
Opponent	Dr. Arto Kaarna LUT University Finland	

The originality of this thesis has been checked using the Turnitin OriginalityCheck service.

Cover design: Roihu Inc.

ISBN 978-952-03-2425-4 (print)
ISBN 978-952-03-2426-1 (pdf)
ISSN 2489-9860 (print)
ISSN 2490-0028 (pdf)
http://urn.fi/URN:ISBN:978-952-03-2426-1

PunaMusta Oy – Yliopistopaino
Joensuu 2022

To my parents **Fu, Jianhua** and **Xia, Qiulian** and my grandma.

PREFACE

This research work was mainly done before 2018 in Tampere, Finland. I would like to express my deep gratitude to my supervisor Professor Joni Kämäräinen for his inspiration, endless support, and encouragement during the entire journey! It has been a great pleasure to meet Associate Professor Said Pertuz during his research visit in Tampere University, and I want to offer my special thanks to him for being my co-supervisor and giving valuable and constructive suggestions during the planning, development, and presentation of a research paper. I am grateful to Professor Jiri Matas for his insightful discussion and passion for research, and I always get motivated after meeting him.

Besides, I want to thank the current and former professors and researchers in the Computer Vision Group for creating such a friendly atmosphere and having many fun activities outside work. Thanks especially to Prof. Heikki Huttunen, Prof. Esa Rahtu, Nataliya Strokina, Ke Chen, Yanlin Qian, Antti Hietanen, Huiling Wang, Antti Ainasoja, Fatemeh Yancheshmeh, Song Yan, Dan Yang, Soumya Tripathy, DingDing Cai, Wenyan Yang, and Lingyu Zhu.

Furthermore, I like to thank Dr. Lixin Fan and Dr. Ville-Veikko Mattila who recruited me to the Multimedia Lab in Nokia Research and introduced me to Computer Vision. I also want to thank my research collaborators, Dr. Yu You and Kimmo Roimela from the team. Thanks to Arto Lehtiniemi, Antti Eronen, Jussi Leppänen, Tinghua Wang, Sujeet Mate, Pouria Babahajiani, and Henri Toukomaa for being great colleagues as well as bringing me to different fun activities, e.g., floorball and ping pong.

Finally, I am grateful to all my family members for their love and unconditional support! I love you all!

ABSTRACT

This PhD thesis targets on two research problems: (1) How to efficiently and robustly estimate the camera pose of a query image with a map that contains street-view snapshots and point clouds; (2) Given the estimated camera pose of a query image, how to create meaningful and intuitive applications with the map data.

To conquer the first research problem, we systematically investigated *indirect*, *direct* and *hybrid* camera pose estimation strategies. We implemented state-of-the-art methods and performed comprehensive experiments in two public benchmark datasets considering outdoor environmental changes from ideal to extremely challenging cases. Our key findings are: (1) the indirect method is usually more accurate than the direct method when there are enough consistent feature correspondences; (2) The direct method is sensitive to initialization, but under extreme outdoor environmental changes, the mutual-information-based direct method is more robust than the feature-based methods; (3) The hybrid method combines the strength from both direct and indirect method and outperforms them in challenging datasets.

To explore the second research problem, we considered inspiring and useful applications by exploiting the camera pose together with the map data. Firstly, we invented a 3D-map augmented photo gallery application, where images' geo-meta data are extracted with an indirect camera pose estimation method and photo sharing experience is improved with the augmentation of 3D map. Secondly, we designed an interactive video playback application, where an indirect method estimates video frames' camera pose and the video playback is augmented with a 3D map. Thirdly, we proposed a 3D visual primitive based indoor object and outdoor scene recognition method, where the 3D primitives are accumulated from the multiview images.

CONTENTS

List of Figures

List of Tables

ORIGINAL PUBLICATIONS

Publication I J. Fu, S. Pertuz, J. Matas and J.-K. Kämäräinen. Performance analysis of single-query 6-DoF camera pose estimation in self-driving setups. *Computer Vision and Image Understanding* 186 (2019), 58–73. DOI: `https://doi.org/10.1016/j.cviu.2019.04.009`.

Publication II J. Fu, L. Fan, K. Roimela, Y. You and V. Mattila. A 3D map augmented photo gallery application on mobile device. *2014 IEEE International Conference on Image Processing (ICIP)*. 2014, 2507–2511. DOI: `10.1109/ICIP.2014.7025507`.

Publication III J. Fu, L. Fan, Y. You and K. Roimela. Augmented and Interactive Video Playback Based on Global Camera Pose. *Proceedings of the 21st ACM International Conference on Multimedia*. 2013, 461–462. DOI: `10.1145/2502081.2502269`.

Publication IV J. Fu, J.-K. Kämäräinen, A. G. Buch and N. Krüger. Indoor Objects and Outdoor Urban Scenes Recognition by 3D Visual Primitives. *Computer Vision - ACCV 2014 Workshops*. Ed. by C. Jawahar and S. Shan. 2015, 270–285. DOI: `https://doi.org/10.1007/978-3-319-16628-5_20`.

Author's contribution

Publication I Junsheng Fu is the main author of this publication. His work covers literature review, experiments design, implementation, and paper writing. The other three coauthors are supervisors for this research work and they contribute to this publication with discussing the research question, giving feedback on experiments and reviewing the writing.

Publication II Junsheng Fu is the main author of this publication. His work covers literature review, experiments design, implementation, paper writing and presentation. The part of 3D scene rendering uses the code developed by coauthor Kimmo Roimela. The data communication between the client and the server uses the code developed by coauthor Dr. Yu You.

Publication III Junsheng Fu is the main author of this publication. His work covers literature review, experiments design, implementation, paper writing and presentation. The part of 3D scene rendering uses the code developed by coauthor Kimmo Roimela. The data communication between the client and the server uses the code developed by coauthor Dr. Yu You.

Publication IV Junsheng Fu is the main author of this publication. His work covers literature review, experiments design, implementation, paper writing and presentation. In the experiment part, the 2D primitives extraction uses the code implemented by coauthor Dr. Anders Glent Buch.

Author's granted patents

All these granted patents were made during the thesis projects and related to the research topic of this PhD thesis, but they are not included as part of this PhD dissertation due to the writing norms of the patent.

1. Junsheng Fu, and Sujeet Shyamsundar Mate. "Method and apparatus for generating a media capture request using camera pose information." *U.S. Patent 9,596,404*, issued March 14, 2017.
2. Lixin Fan, Junsheng Fu, Kimmo Tapio Roimela, and Yu You. "Method and apparatus for determining camera location information and/or camera pose information according to a global coordinate system." *U.S. Patent 9,699,375*, issued July 4, 2017.
3. Lixin Fan, Junsheng Fu, Kimmo Roimela, and Yu You. "Method and apparatus for determining camera location information and/or camera pose information according to a global coordinate system." *U.S. Patent 9,558,559*, issued January 31, 2017.
4. Lixin Fan, Ville-veikko Mattila, Yu You, Kimmo Roimela, Junsheng Fu, and Antti Eronen. "Method and technical equipment for determining a pose of a device." *U.S. Patent 10,102,675*, issued October 16, 2018.
5. Sujeet Shyamsundar Mate, Jussi Leppanen, Junsheng Fu, and Pouria Babahajiani. "Device with an adaptive camera array." *U.S. Patent 9,996,934*, issued June 12, 2018.

1 INTRODUCTION

1.1 Background and motivation

The aim of localization is to determine the position and orientation of someone or a object with respect to a reference map. Humans have invented some tools to perform localization, such as the compass that was invented more than 2,000 years ago. We can use a compass and a map to localize ourselves. Let us take Fig. 1.1 as a simplified example: firstly, we visually identify a landmark A which is a peak in a mountain range, so that we can easily find it in the map. Secondly, we measure the orientation of the landmark A with respect to our position by a compass and the map. Thirdly, we draw a line through the landmark A with the observed orientation. Finally, we repeat it with another landmark B, and the crossed point is our location in the map. The process of finding where we are can be called *self-localization*, and the method of pinpointing our location by taking bearings to it from two remote points is called *triangulation*. In the above example, the localization is performed in a 2D case. However, we can get even more detailed location and orientation information with a camera sensor by using the computer vision techniques.

With the increased popularity of mobile camera phones and emergence of smart glasses [89], camera sensors have become one of the most ubiquitous sensors and billions of images are generated in people's everyday life. *Visual localization* is to perform the localization task by estimating the *camera pose* of a given image. The camera pose of an image describes the orientation and location of the camera in a reference coordinate system. The camera pose has up to 6-degrees-of-freedom (DoF), because a camera can rotate in 3-DoF and translate in 3-DoF in a Cartesian coordinate system [48]. Fig. 1.2 illustrates the concept of camera pose. Since both the location and orientation are relative terms, a reference coordinate system is needed to define the camera pose with a 3-DoF orientation and a 3-DoF location.

What are the benefits of knowing the *camera pose* of a given image? Camera pose

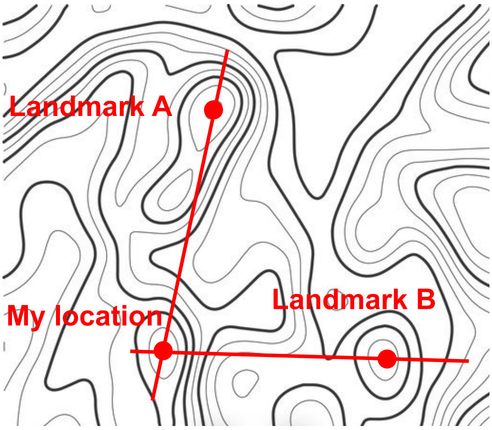


Figure 1.1 Localize oneself with a compass and a map. Draw a line through each landmark with the observed orientation by the compass, and the crossed point is the estimated location.

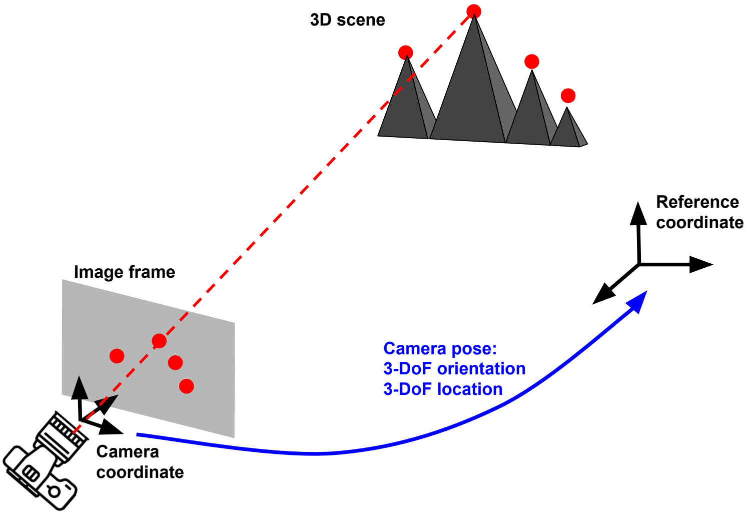


Figure 1.2 The camera takes an image of a 3D scene, and its camera pose is defined as a 3-DoF orientation and a 3-DoF location with respect to a predefined reference coordinate system.

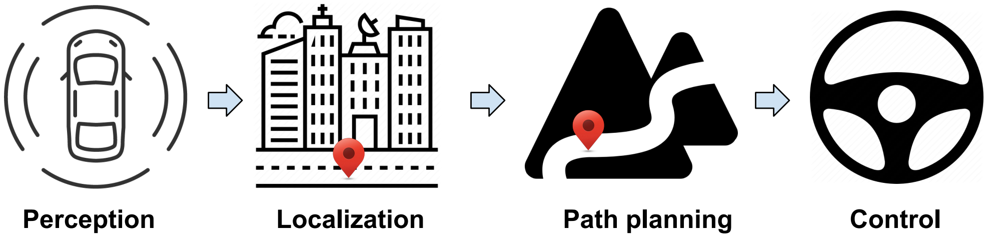


Figure 1.3 Four major components for an autonomous vehicle software system.

estimation can be an enabling technology for autonomous robots [64, 107], augmented reality [89, 99], virtual reality [7, 14], mixed reality [7], image sharing service [36, 37] and simultaneous localization and mapping (SLAM) [24, 25, 98, 139]. Let us take a look how the camera pose estimation contributes to the 3 following applications.

Application 1: Autonomous vehicles. In recent years, major automakers, automative suppliers, tech giants, start-ups and research groups are involved in the research and development of the self-driving car technologies. There are different sensor setups for autonomous vehicles and there has been continous debate regarding the best sensor set for autonomous vehicles [132]. However, cameras are always used in all sensor setups from companies. In an autonomous driving system (ADS), there are four major software components: *perception*, *localization*, *path planning*, and *control*, as shown in Fig. 1.3. In the *perception* stage, the ADS senses surroundings with various sensors and understands what the world looks like, e.g. where are the lanes, other vehicles, pedestrians, etc. In the *localization* stage, the ADS figures out its own pose in the world, i.e. determines the location and orientation of itself in a world coordinate system. Once the ADS knows its own pose and where other objects are in the world, in the *path planning* stage, ADS plans a safe path for vehicle to drive. Finally, in the *control* stage, ADS steers the car and engages the throttle or the brake to follow that planned path in a safe manner.

The second component, *localization*, can be achieved by several approaches, and *camera pose estimation* is one method to perform the *localization* task for the autonomous driving system. One example is shown in Fig. 1.4, the camera pose corresponding to a query image is estimated with a given reference image and a corresponding point cloud. The query image can be viewed as the current camera input,

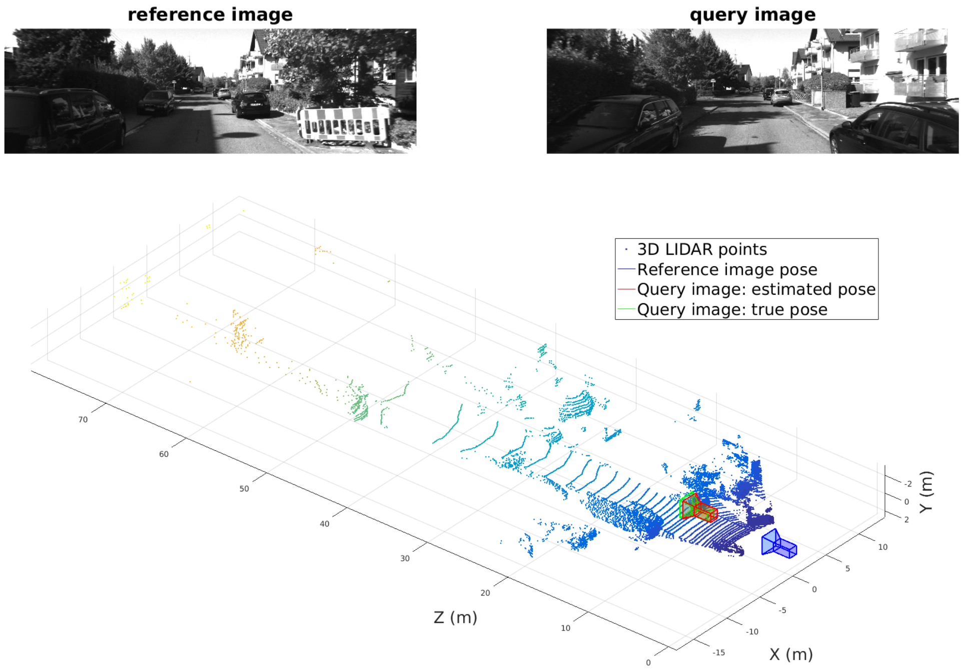


Figure 1.4 Camera pose estimation in a self-driving car setup. The top two images are used as the query and reference images from the KITTI dataset [42].The bottom figure illustrates the estimated 6-DoF camera pose of the query image with respect to the reference image and a 3D point cloud.

and the reference image together with the point cloud can be considered as a map. The task is given the current camera view and a map, the ADS estimates the 6-DoF pose of the vehicle in the map.

Application 2: Image sharing service. With the increasing ownership rate of mobile phone cameras and popularity of various social media apps, capturing images with a camera phone and sharing photos with friends on social media are gradually becoming parts of our daily activities. Do you have the experience of seeing an interesting photo of a tourist attraction either from your friends or online, and you would like to see more of the surroundings? For example, you would like to see what is on the left, what is on the right or even what is on the opposite side of the captured scene. This can be done by utilizing the *camera pose estimation* and a 3D map. A 3D map refers to a map of textured 3D models.

With the estimated camera pose, we know exactly where the image was captured in the 3D map. Then, the user gains the possibility to explore the surroundings by utilizing the camera pose together with the point of interest information in the 3D map. The camera pose of the captured image can be seamlessly visualized with the surrounding 3D environment. Furthermore, exploiting the estimated camera pose, users' view can transit from 2D image view to 3D map space. If the user would like to explore more of the surroundings, the user could even change their view angles of the scene by navigating in the 3D map. Therefore, the ordinary image capturing and sharing experience can be greatly enriched by leveraging the estimated camera pose. One interactive image sharing application developed by the author can be seen in this video [1].

Application 3: Mixed reality and augmented reality. Mixed reality is the merging of virtual and real worlds to produce new visualizations and environments, where physical and digital objects co-exist and interact in real time [7]. With the emergency of smart glasses, such as Microsoft Hololens [89] and Google-glass [45], augmented reality gained attention in the gaming industry [80] and education [9]. Besides, smart glasses could also assist in discovering the surroundings through an augment reality App [89].

One key challenge for both mixed reality and augmented reality is how to efficiently compute the camera pose. The traditional real-time approach is the inertial sensor-based approach, e.g. GPS and IMU, but the purely inertial sensor-based approaches can be sensitive to environmental noise. Image-based camera pose estimation can strengthen the sensor-based approach, and visual-inertial camera pose estimation is a another popular way to compute the real-time device pose [56, 129]. Once the camera pose is successfully estimated, many intuitive mixed reality applications become possible. One intuitive augmented reality application for video playback is developed by the author and it can be seen here[2].

1.2 Objective of the thesis

My research aims to find efficient and robust approaches of camera pose estimation with street-view snapshots and 3D point clouds, and experiment on the usage of the estimated camera pose in different applications.

[1] https://junshengfu.github.io/videos/3D_photo_Album/3DPhotoAlbum.mp4

[2] https://junshengfu.github.io/videos/video_playback/videoPlayBack.mp4

1.3 Summary of the original articles

Publication 1 considers the problem of single-query 6-DoF camera pose estimation, i.e. estimating the position and orientation of a camera by using reference images and a point cloud. We perform a systematic comparison of three state-of-the-art strategies for 6-DoF camera pose estimation: feature-based, photometric-based and mutual-information-based approaches. Two standard datasets with self-driving setups are used for experiments, and the performance of the studied methods is evaluated in terms of success rate, translation error and maximum orientation error. Building on the analysis of the results, we evaluate a hybrid approach that combines feature-based and mutual-information-based pose estimation methods to benefit from their complementary properties for pose estimation. Experiments show that (1) in cases with large appearance change between query and reference, the hybrid approach outperforms feature-based and mutual-information-based approaches by an average increment of 9.4% and 8.7% in the success rate respectively; (2) in cases where query and reference images are captured at similar imaging conditions, the hybrid approach performs similarly as the feature-based approach, but the hybrid approach outperforms both photometric-based and mutual-information-based approaches with a clear margin; (3) the feature-based approach is consistently more accurate than mutual-information-based and photometric-based approaches when enough consistent matching points are found between the query and reference images.

Publication 2 presents a 3D map augmented photo gallery mobile application that utilizes the estimated camera pose together with the 3D Map data to enrich the ordinary photo sharing and browsing applications. This mobile application has a client-server architecture, and automatically computes the global camera pose of the user captured images. With the mobile rendering, users can seamlessly transit their views from 2D images to 3D map, navigate in the 3D map, and even travel to the nearby scenes which are not visible in the original image.

Publication 3 proposes a client-server architecture that uses geo-metadata to enrich the video playback experience. In contrast to the existing video playback systems, our system computes the global 6-DoF camera pose of the video frames, and allow users to expand the field of view, see the 6-DoF camera motion, and arbitrarily change the viewing angle. The ordinary video capturing and sharing experience can be greatly enriched by leveraging geo-metadata associated with video frames.

Publication 4 proposes a 3D visual primitives-based recognition method, which utilizes both 2D appearance and 3D structure from multi-view images. The low level 2D visual primitives are categorized by computational intrinsic dimension, and then they are matched across multi-view images and accumulated as 3D primitives. A simple but effective RANSAC variant is introduced to match the 3D primitives. Experimental results show that the process of accumulation from 2D primitives to 3D primitives improves the object recognition accuracy by selecting more robust primitives. The 3D primitives-based approach is more robust for viewpoint changes compared with 2D primitives-base approach. Our proposed method achieved good accuracy for the view angle variation up to ±20° with indoor objects dataset and satisfactory accuracy for the urban street-view scenes.

1.4 Outline of the thesis

This thesis is divided into 7 chapters. Chapter 1 gives a background introduction and motivation for the PhD research topic. Chapter 2 introduces the fundamental concepts and the main datasets used in this thesis. Chapter 3 presents the detailed literature review for camera pose estimation methods. Chapter 4 describes single-query 3D camera pose estimation. It addresses various approaches, create test benches, evaluate them quantitatively and propose a hybrid approach. Chapter 5 proposes two innovative augmented reality applications by using camera pose and a 3D map. Chapter 6 designs and quantitatively evaluates a 3D method for place and object recognition. Chapter 7 concludes the thesis.

2 FUNDAMENTALS

Camera sensor is the main sensor used in this thesis for the camera pose estimation problem. Besides the camera sensor, the Light-Detection-and-Ranging (LIDAR) sensor is used in several public datasets, where LIDAR data are treated as 3D reference data. In this Chapter, we want to establish some vocabularies for describing the image formation process of a camera and the main principles for a LIDAR. Furthermore, we explain the definition of the *street-view snapshots* and the *Point cloud* in this thesis. Last but not least, we present the main datasets used in this thesis.

2.1 Image formation

2.1.1 How a camera works

This Section briefly presents how a camera works. We describe how a ray of light reflected from an object passes through a lens and aperture, lands on photosensitive sensor surface, and then is converted to discrete color value.

1. The Pinhole camera is a simple camera model that can be designed by placing a light barrier with a tiny hole between an object of interest and a photosensitive surface. The light reflected from the object passes through the pinhole and lands on a photosensitive surface which stores the light information as an image. A pinhole camera is illustrated in Fig. 2.1a. The photosensitive surface is the image plane, the distance between the pinhole and the image plane is the focal length, and the pinhole is considered as the camera origin. The pinhole is very small and limits the amount of light that can pass through. To allow more light pass through, a straightforward way is to enlarge the pinhole, but the light would overlap on the image plane resulting in image blur as shown in Fig. 2.1b. To solve this problem, a camera lens and a aperture can be used.

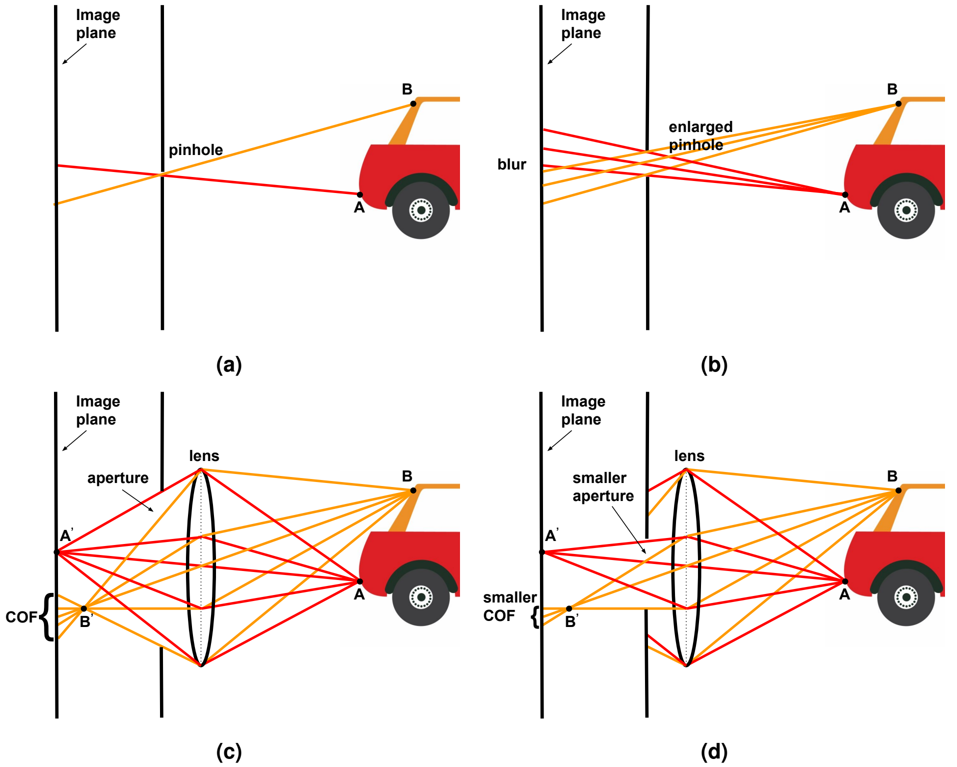


Figure 2.1 (a) A pinhole camera. (b) An enlarged pinhole allows more light into image plane, but it introduces the image blurring effect. (c) Placing a lens in front of a camera makes the rays from a point of an object converge to a single point. However, it brings blurring effect too. (d) The blurring effect can be reduced by adjusting the aperture size.

2. By placing a suitable lens in front of a camera, all rays of light from a point of an object converge to a single point. For example, all rays of light from point A in the bottom converge to A' on the image plane as plotted in Fig. 2.1c. However, not all light from a source point can land on the exact same spot on image plane. Let's take the point B in Fig. 2.1c as an example, B' lands before the image plane, so the point B is visualized as a blurred spot and the blurry circle is named as circle of confusion (COF) [113]. To reduce the blurring effect, the aperture can be adjusted as shown in Fig. 2.1d.

3. Once the light passes through the lens and lands on the photosensitive sensor surface, the amount of light is counted and converted into a corresponding electric charge. Aperture and exposure time can be used to adjust the amount of light that can pass through. As we can imagine, the more light generates

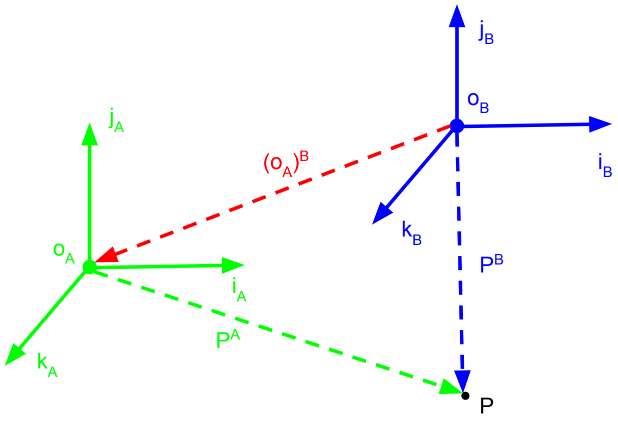


Figure 2.2 The 3D translation of 2 Cartesian coordinate systems with parallel axes can be seen as a vector $(\mathbf{o}_A)^B$ between two origins.

more electrons and vice versa. Once the shutter is closed, the produced electrons are converted into a voltage and the voltage is transformed into a discrete number by an A/D-converter. Usually, a color imaging array filter is placed before the photosensitive surface to enable the color vision.

2.1.2 Camera extrinsics and camera pose

A 3D Cartesian coordinate system consists of an origin point and three mutually perpendicular unit vectors (i, j, k) which define 3 coordinate axes. The 3D Cartesian coordinate system is widely used in camera pose estimation and its related applications.

Camera extrinsic parameters define the coordinate system transformations from a world coordinate system to the camera's 3D Cartesian coordinate system. Camera extrinsics consist of *translation* and *rotation*.

2.1.2.1 Translation

Let us assume two 3D Cartesian coordinate systems A and B as illustrated in Fig. 2.2, and they have parallel corresponding axes, i.e. i_A is parallel to i_B, j_A is parallel to j_B, and k_A is parallel to k_B. Take an arbitrary 3D point $\mathbf{p}$, and $\mathbf{p}$ can be expressed in both coordinate systems. $\mathbf{p}^A$ is the 3D coordinates of $\mathbf{p}$ in frame A, $\mathbf{p}^B$ is the 3D coordinates of $\mathbf{p}$ in frame B, and $(\mathbf{O}_A)^B$ is the frame A's origin $\mathbf{O}_A$ represented in frame B. $\mathbf{p}^A$, $\mathbf{p}^B$ and $(\mathbf{O}_A)^B$ should satisfy the following equation:

$$\mathbf{p}^B = \mathbf{p}^A + (\mathbf{O}_A)^B \tag{2.1}$$

Equation (2.1) can be seen as vector addition in 3D space. $\mathbf{p}^A$ could be seen as a vector starting from origin $\mathbf{O}_A$ and pointing to $\mathbf{p}$ and $\mathbf{p}^B$ could be viewed as a vector starting from origin $\mathbf{O}_B$ and pointing to $\mathbf{p}$. $(\mathbf{O}_A)^B$ is a vector starting from origin $\mathbf{O}_B$ and pointing to $\mathbf{O}_A$, and $(\mathbf{O}_A)^B$ is essentially the *translation* between frames. The *translation* vector can be expressed as:

$$(\mathbf{O}_A)^B = \mathbf{p}^B - \mathbf{p}^A \tag{2.2}$$

Using the homogeneous coordinates, the equation (2.1) can be expressed as a matrix multiplication:

$$\begin{bmatrix} \mathbf{p}^B \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{I} & (\mathbf{O}_A)^B \\ \mathsf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p}^A \\ 1 \end{bmatrix} \tag{2.3}$$

where $\mathbf{I}$ is a 3×3 identity matrix and $\mathsf{0}^T$ is a 1×3 zero vector.

2.1.2.2 Rotation

Two 3D Cartesian coordinate systems A (green) and B (blue) share the same origin in Fig.2.3, and each coordinate system is represented by three mutually orthogonal unit vectors i, j, k. A 3D point $\mathbf{p}$ can be expressed in both A and B coordinate systems, as follows:

$$\overrightarrow{OP} = \begin{bmatrix} \mathbf{p}_x{}^A & \mathbf{p}_y{}^A & \mathbf{p}_z{}^A \end{bmatrix} \begin{bmatrix} i_A \\ j_A \\ k_A \end{bmatrix} = \begin{bmatrix} \mathbf{p}_x{}^B & \mathbf{p}_y{}^B & \mathbf{p}_z{}^B \end{bmatrix} \begin{bmatrix} i_B \\ j_B \\ k_B \end{bmatrix} \tag{2.4}$$

Given two frames A and B as illustrated in Fig.2.3, there is an existing relation between $\mathbf{p}^A$ $(\mathbf{p}_x{}^A, \mathbf{p}_y{}^A, \mathbf{p}_z{}^A)$ and $\mathbf{p}^B$ $(\mathbf{p}_x{}^B, \mathbf{p}_y{}^B, \mathbf{p}_z{}^B)$ and this relation is called *Rotation*. By applying the rotation, the representation of a point $\mathbf{p}$ in frame A can be trans-

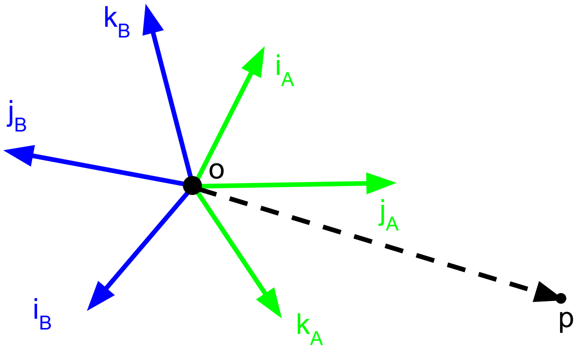


Figure 2.3 3D rotation between 2 Cartesian coordinate systems.

formed to frame B, as shown below:

$$\mathbf{p}^B = \mathbf{R}_A^B \cdot \mathbf{p}^A \tag{2.5}$$

This above equation tells that a point $\mathbf{p}$ described in frame A can be expressed in frame B by a rotation operator $\mathbf{R}_A^B$. The $\mathbf{R}_A^B$ describes frame A in the coordinate system of frame B, and it can be expressed as:

$$\mathbf{R}_A^B = \begin{bmatrix} i_A \cdot i_B & j_A \cdot i_B & k_A \cdot i_B \\ i_A \cdot j_B & j_A \cdot j_B & k_A \cdot j_B \\ i_A \cdot k_B & j_A \cdot k_B & k_A \cdot k_B \end{bmatrix} \tag{2.6}$$

Each column of the equation (2.6) expresses how each basis vector in frame A is expressed in frame B. By looking at the 1st column, $i_A \cdot i_B$ means how the i vector in frame A is expressed in terms of magnitude in the i direction of frame B. Similarly, $i_A \cdot j_B$ means how the i vector in frame A is expressed in terms of magnitude in the j direction of frame B, and $i_A \cdot k_B$ means how the i vector in frame A is expressed in terms of magnitude in the k direction of frame B. Therefore, the 1st column could be considered as how the vector i_A is expressed in i, j, k directions in frame B. Similarly, the 2nd and 3rd columns indicate how the vector j_A and k_A are expressed in i, j, k directions in frame B. As a result, the rotation $\mathbf{R}_A^B$ expressed in equation (2.6) could be written as:

$$\mathbf{R}_A^B = \begin{bmatrix} i_A^B & j_A^B & k_A^B \end{bmatrix} \tag{2.7}$$

If we look at the 1st row of equation (2.6), it actually tells how the i vector in frame B is expressed in i, j, k directions in frame A. Similarly, 2nd and 3rd rows tell how the j_B and k_B vectors are expressed in i, j, k directions in frame A. Therefore, the rotation $\mathbf{R}_A^B$ expressed in equation (2.6) could be written as:

$$\mathbf{R}_A^B = \begin{bmatrix} {i_B^A}^T \\ {j_B^A}^T \\ {k_B^A}^T \end{bmatrix} \tag{2.8}$$

If we take the transpose of equation (2.8), we get following equation (2.9),

$$\begin{aligned} (\mathbf{R}_A^B)^T &= \begin{bmatrix} {i_B^A}^T \\ {j_B^A}^T \\ {k_B^A}^T \end{bmatrix}^T \\ &= \begin{bmatrix} i_B^A & j_B^A & k_B^A \end{bmatrix} = \mathbf{R}_B^A \end{aligned} \tag{2.9}$$

Therefore, the inverse of a rotation matrix is its transpose.

The rotation operation could also be expressed with the homogeneous coordinates, and the equation (2.5) can be written as:

$$\begin{bmatrix} \mathbf{p}^B \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_A^B & 0 \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p}^A \\ 1 \end{bmatrix} \tag{2.10}$$

2.1.2.3 Camera pose

Camera pose defines the *orientation* and *location* of a camera in a world coordinate system. The camera pose has a maximum of 6 degree-of-freedom, and it can be represented as 3 Euler angles (α, β, γ) and 3 location components (x_0, y_0, z_0) in axes (x, y, z) respectively. Given a camera pose, we could obtain the a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$ describing a coordinate transform from the camera coordinate to the world coordinate as follows:

$$\begin{aligned}\mathbf{R} &= \mathbf{R}_z(\gamma)\mathbf{R}_y(\beta)\mathbf{R}_x(\alpha)\\ &= \begin{bmatrix} cos(\gamma) & -sin(\gamma) & 0\\ sin(\gamma) & cos(\gamma) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix} cos(\beta) & 0 & sin(\beta)\\ 0 & 1 & 0\\ -sin(\beta) & 0 & cos(\beta)\end{bmatrix}\begin{bmatrix} 1 & 0 & 0\\ 0 & cos(\gamma) & -sin(\gamma)\\ 0 & sin(\gamma) & cos(\gamma)\end{bmatrix}\end{aligned} \tag{2.11}$$

$$\mathbf{t} = \begin{bmatrix} x_0\\ y_0\\ z_0\end{bmatrix} \tag{2.12}$$

The definitions of the camera extrinsic and the camera pose are in opposite coordinate systems, because camera extrinsic describes a coordinate transforms from a world coordinate system to camera coordinate system and camera pose describes rotation and translation from camera coordinate system to a world coordinate system. Therefore, given $\mathbf{R}$ (2.11) and $\mathbf{t}$ (2.12) from a 6-DoF camera pose $(\alpha, \beta, \gamma, x_0, y_0, z_0)$, the *camera extrinsic homogeneous matrix* $\mathbf{T}$ could be expressed as:

$$\begin{aligned}\mathbf{T} &= \left(\begin{bmatrix} \mathbf{R} & \mathbf{t}\\ 0^T & 1\end{bmatrix}\right)^{-1}\\ &= \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T\mathbf{t}\\ 0^T & 1\end{bmatrix}\end{aligned} \tag{2.13}$$

By applying camera extrinsic matrix $\mathbf{T}$ (2.13) to a 3D point $\mathbf{p}$ defined in a world coordinate system, we could obtain the 3D coordinate of the same point defined in the camera coordinate system, as shown in equation (2.14).

$$\begin{bmatrix} \mathbf{p}^{'}\\ 1\end{bmatrix} = \mathbf{T}\begin{bmatrix} \mathbf{p}\\ 1\end{bmatrix} \tag{2.14}$$

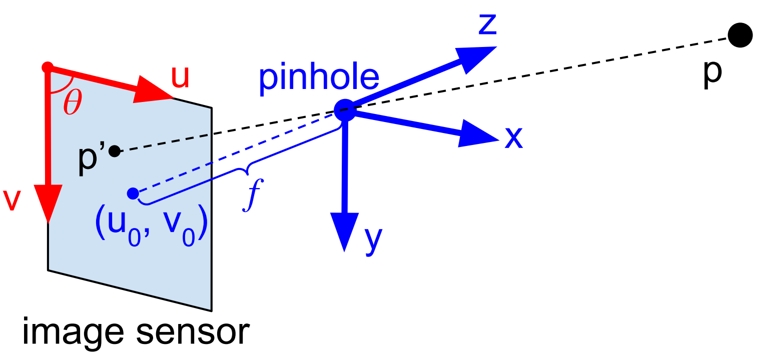


Figure 2.4 Illustrate the focal length, principal point and axes skew in a pinhole camera model.

2.1.3 Camera Intrinsics

Camera intrinsic parameters define coordinate transforms from 3D camera coordinates to 2D image coordinates, and the camera intrinsic matrix **K** can be expressed as below [48]:

$$\mathbf{K} = \begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.15}$$

Let us take Figure 2.4 as an example, and go through each of these parameters, i.e. focal length (f_x, f_y), principal point (u_0, v_0) and axes skew (s).

2.1.3.1 Focal length

The focal length f is the distance between the camera pinhole to the image sensor as shown in Fig. 2.4, and it is usually measured in millimeters. However, in order to represent u and v in pixels, the focal length f is expressed in pixels as f_x and f_y. In reality, the width of a pixel may not be exactly the same as the height of pixels, i.e. non-square pixels. Therefore, f_x and f_y in equation (2.15) are the focal lengths expressed in width and height of the pixels. If w and h are the width and the height of the pixels in millimeters, then $f_x = \frac{f}{w}$ and $f_y = \frac{f}{h}$.

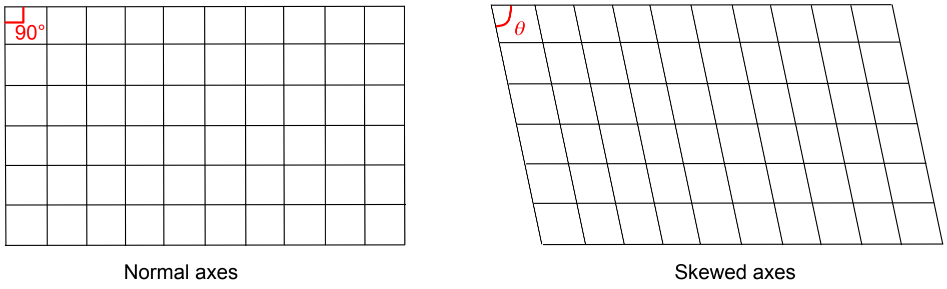


Figure 2.5 Compare skewed coordinates with normal coordinates.

2.1.3.2 Principal point

In pinhole camera model, the *principal axis* is the line perpendicular to the image plane which passes through the camera pinhole. Its intersection with the image sensor is called *principal point*. In equation (2.15), u_0 and v_0 are the position of the principal point. Fig. 2.4 illustrates u_0 and v_0. A the projected point in image plane can be expressed as:

$$\begin{aligned} u &= f_x \frac{x}{z} + u_0, \\ v &= f_y \frac{y}{z} + v_0 \end{aligned} \tag{2.16}$$

2.1.3.3 Axes skew

Axes skew is caused by the two axes of the image sensor not being perpendicular to each other, and it is shown in Figure 2.5.

After superimposing both the skewed axes and the normal axes, we can use trigonometry to find the coordinates of a point in skewed frame (x_{skew}, y_{skew}) in terms of its coordinates in normal frame (x, y), shown as in Figure 2.6:

$$\begin{aligned} x_{skew} &= x - ycot(\theta), \\ y_{skew} &= \frac{y}{sin(\theta)} \end{aligned} \tag{2.17}$$

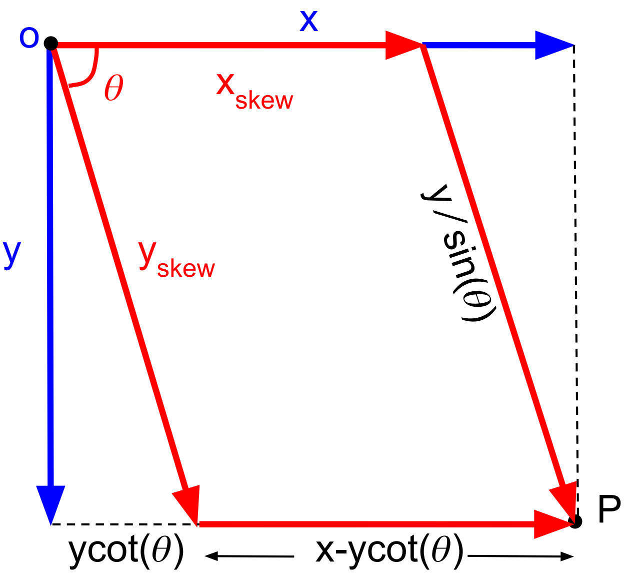


Figure 2.6 Superimposing both skewed axes (x_{skew}, y_{skew}) and the normal axes (x, y).

Since the focal length (f), width of pixel and depth of the object (z) remain unchanged, the equation (2.17) can be adapted for the skewed coordinates.

$$\begin{aligned} u &= f_x \frac{x_{skew}}{z} + u_0, \\ v &= \frac{f_y}{sin(\theta)} \frac{y_{skew}}{z} + v_0 \end{aligned} \tag{2.18}$$

Then, substituting equation (2.17) into the equation (2.18), we get

$$\begin{aligned} u &= \frac{f_x}{z} x - \frac{f_x}{z} cot(\theta) y + u_0, \\ v &= \frac{f_y}{z sin(\theta)} y + v_0 \end{aligned} \tag{2.19}$$

If we multiply equation (2.19) with z in both sides, this equation can be written as,

$$
\begin{aligned}
\begin{bmatrix} zu \\ zv \\ z \end{bmatrix} &= \begin{bmatrix} f_x & -f_x cot(\theta) & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\
&= \begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\
&= K \begin{bmatrix} x \\ y \\ z \end{bmatrix}
\end{aligned}
\tag{2.20}
$$

where s is usually used as the skew parameter and K is the camera intrinsic matrix. However, the skew parameter of the most cameras is usually zero nowadays.

2.2 Principle of LIDAR

LIDAR is short for Light Detection And Ranging, and the idea behind LIDAR system has existed in nature for a long time. Several animal species, such as bats, dolphins, whales, utilize echoes to determine where objects are in the space, and this type of guidance system is called *echolocation*. Take bats as an example, bats could emit short and loud chirps, and when the sound waves hit objects echoes are produced. Bats use their ears to receive the echo bounces off the objects, so they can determine the size and shape of the objects in their surroundings. LIDAR works in the similar principle, but instead of sending out sound wave LIDAR sends out laser beams.

LIDAR produces a high accurate 3D model of the surrounding environment. If we take a basic spindle-type LIDAR as an example, it has 4 main components, Laser emitters, Laser receivers, motor that spins to give 360-degree view, and a mounting base. The general process of a LIDAR sensor is described as follow.

1. LIDAR sends out infrared laser beams at different angles and starts timing.

Some laser beams hit the surface of the objects and reflect.

2. Some reflected laser beams are detected by a receiver in LIDAR, and LIDAR can calculate the distance of the obstacles by measuring the time that laser beam traveled to and back from the obstacles. The distance can be expressed as: $d = \frac{1}{2} \times c \times t$, where c is speed of light and t is the time of flight.
3. From the received intensity values of the returned laser beams, it is also possible to estimate the surface material properties of the obstacles.

One example of the LIDAR sensors is Velodyne HDL-64E and it is used in KITTI dataset [42]. This LIDAR can sense objects up to 120 meters and create up to 2.2 million 3D points per second.

2.3 Street-view snapshots and point clouds

For an given outdoor image, the *street-view snapshots* and *point clouds* are used as the main references to estimate the camera pose in this Thesis work. Fig. 2.7 shows examples of a street-view snapshot and a LIDAR Point Cloud.

Street-view snapshots in this thesis refer to the images that are captured by the cameras mounted in a mapping vehicle. Some of the experiments in this thesis take the street-view snapshots in the public datasets [42, 82] as the reference images for evaluating different camera pose estimation methods.

Point clouds are a collection of 3D points that represent the 3D model of the surroundings. Usually, point clouds can be created by two approaches, i.e., the LIDAR system and structure from motion techniques [124, 127, 128, 145, 146].

1. The LIDAR system is discussed in the Section 2.2, and four point cloud processing techniques have been used in this Thesis. (1) The point cloud accumulation is used to build up a bigger point cloud from a sequential timestamps based on the given relative sensor pose parameters, and it is a useful technique if the LIDAR system has a small detection range. The relative pose parameters between two timestamps are given in the dataset, because the LIDAR point clouds are used as the reference in our experience and the ground truth pose for every timestamp is provided. (2) The point cloud down-sampling is a technique used to reduce the number of points in a point cloud and deal

with the outliers [47]. The voxel grid filtering [119] is a popular point cloud down-sampling method. It creates a cubic grid over the input point cloud and then the points in each voxel cubic are approximated with their centroid, so the larger the cube length the lower the resolution of the output point cloud [119]. (3) The region of interest is a defined boxed region and any points outside the box will be removed [119]. (4) The hidden point removal [61] is a technique that filters out the points that can not be seen by the current view if the point cloud was a closed surface.

2. The structure from motion (SfM) techniques [127, 128, 140] aim to estimate the 3D structure of a scene and camera poses from a set of 2D images. The simplest case of SfM is 2 images from either 2 stationary cameras or one moving camera, and it consists of 4 main steps: (1) It computes the point correspondences between two images with feature matching techniques [33, 77, 94]. (2) It estimates the relative pose of the second view with regarding to the first view. (3) It finds point tracks across the two views, and uses triangulation [48] to compute their initial 3D positions. (4) It uses bundle adjustment [48, 146] to refine the camera poses and 3D points. Moreover, the SfM approach for two views can be extended for multiple views by transforming all camera poses into a common coordinate system, then it applies the bundle adjustment. Incremental SfM is an approach that adds on one image at a time to grow the reconstruction [124]. While this type of method is robust, it requires repeated operations of the expensive bundle adjustment. Therefore, some improved bundle adjustment strategy [93, 145] are proposed to provide better balance between speed and accuracy. It is worth noting that if the images are taken with a single calibrated camera, then the reconstructed 3D scene and camera motion can only be recovered up to scale. To estimate the actual scale of the 3D scene and camera motion in world units, we need to know the actual size of an object in the scene or some additional sensor inputs such as an odometer. As a result, the SfM system outputs the 3D point cloud of the scene and camera poses of the inputs images.

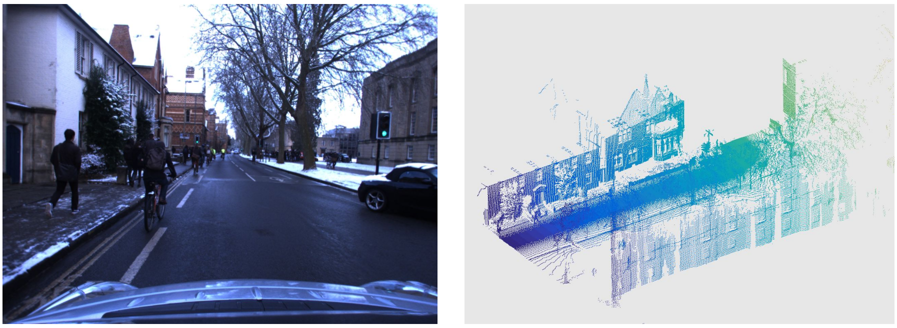


Figure 2.7 A street-view snapshot (left) and a LIDAR point cloud (right) from the Oxford RoboCar Dataset [82].

2.4 Main datasets

2.4.1 Oxford RobotCar

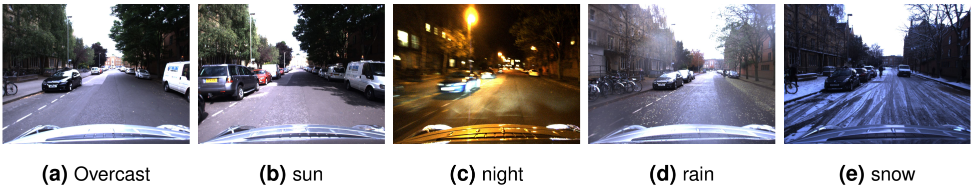
The Oxford RobotCar dataset [82] provides multiple repetitions of a consistent route through Oxford in the UK over the period of one year. It contains different combinations of illumination, weather, traffic and road conditions. In our experiments, we selected 5 sequences with completely different environmental conditions, i.e. snow, rain, night, sunny, overcast. For each sequence, we use (1) point clouds, (2) street-view snapshots and (3) ground-truth pose. The point cloud is captured by a LIDAR sensor. Street-view snapshots are captured by a Bumblebee XB3 camera, and the left images from the Bumblebee XB3 camera are used. The overview of these 5 sequences is summarized in Table 2.1 and example images are shown in Fig. 2.8. This dataset is used in Publication I, and the query and reference data are taken from different sequences in the related experiments. Therefore, it enables a challenging evaluation for different camera pose estimation methods in realistic conditions.

2.4.2 KITTI

The KITTI dataset [42] captures the data around a mid-size city Karlsruhe in Germany including urban areas, rural areas and highways. Similar as the Oxford RoboCar dataset, we use (1) point clouds, (2) street-view snapshots and (3) ground truth

Table 2.1 Overview of 5 sequences with different environmental conditions in Oxford RobotCar dataset [82].

id	# images	tag	total length (km)	mean distance between consecutive images (m)
00	1916	overcast	6.3	3.3
01	2873	sun	8.6	3.0
02	2931	night	9.1	3.1
03	2614	rain	8.8	3.4
04	3019	snow	8.7	2.9



(a) Overcast **(b)** sun **(c)** night **(d)** rain **(e)** snow

Figure 2.8 Images from 5 sequences with different weather conditions in the Oxford RobotCar dataset [82].

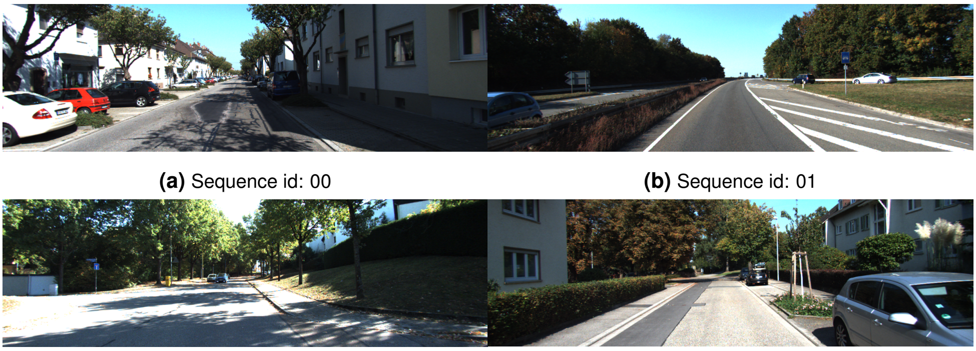
pose from each sequence.The point clouds are captured by a LIDAR sensor. We use images of one gray-scale camera as the street-view snapshots in our experiments. KITTI dataset provides 11 sequences with ground truth and we use all these 11 sequence in our camera pose estimation experiments. The overview of these 11 sequences are summarized in Table 2.2 and example images are shown in Fig. 2.9. This dataset is used in the experiments of Publication I.

2.4.3 HERE

The author got access to the HERE map data [10] for research purpose during the period of working in Nokia Research Center in Finland. Street-view snapshots and their corresponding ground-truth poses are used in the experiments of Publications II, III and IV. The utilized street-view snapshots cover a few different cities, e.g. Helsinki, Tampere, Paris, etc. The examples of street-view snapshots from Here map can be seen in Fig. 2.10.

Table 2.2 Overview of the 11 sequences in the KITTI dataset [42].

id	# images	tag	total length (km)	mean distance between consecutive images (m)
00	4541	urban	3.7	0.8
01	1101	highway	2.5	2.2
02	4661	urban	5.1	1.1
03	801	urban	0.6	0.7
04	271	urban	0.4	1.5
05	2761	urban	2.2	0.8
06	1101	urban	1.2	1.1
07	1101	urban	0.7	0.6
08	4071	urban	3.2	0.8
09	1591	urban	1.7	1.1
10	1201	urban	0.9	0.8



(a) Sequence id: 00

(b) Sequence id: 01

(c) Sequence id: 02

(d) Sequence id: 03

Figure 2.9 Images from four sequences in the KITTI dataset [42].

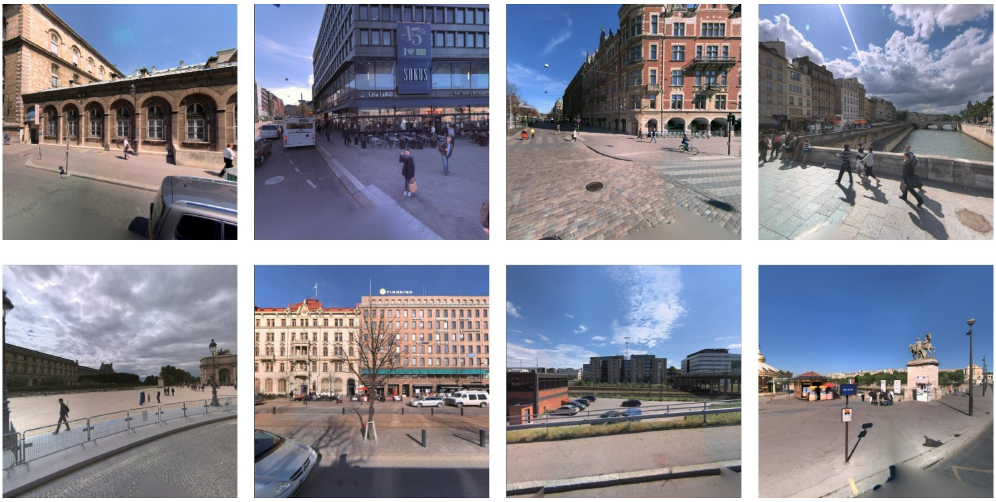


Figure 2.10 Eight examples of HERE street-view snapshots.

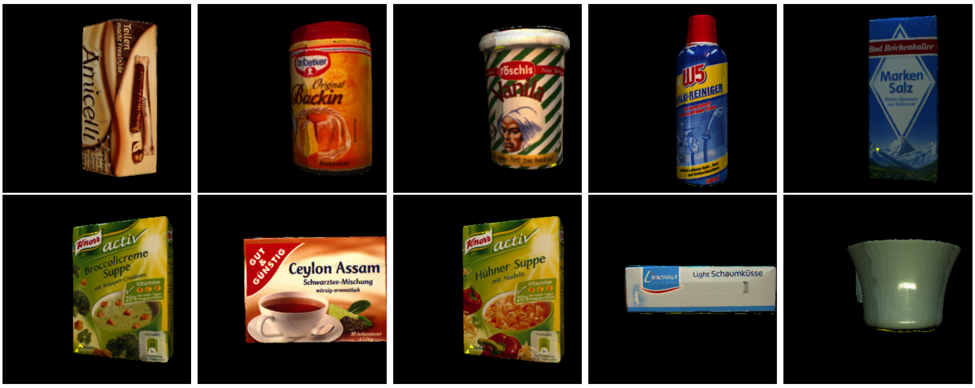


Figure 2.11 Ten examples from KIT object models dataset.

2.4.4 KIT object models

The KIT object models dataset [60] provides high-quality textured 3D models. The 3D models are mainly kitchen items, e.g. mugs, tea packages, bottles, shown as in Fig. 2.11. To create the 3D models, objects are placed on a turning able and then they are scanned with a laser scanner to obtain the point clouds. The point clouds from different views are post-processed to produce the 3D models. This KIT dataset is used in Publication IV.

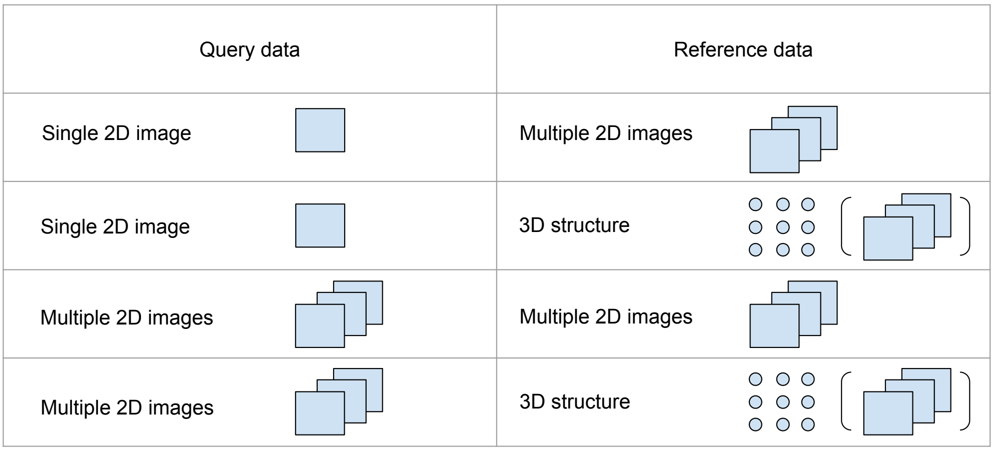
3 LITERATURE REVIEW

Camera pose estimation is an enabling technology for various applications, e.g. robotic localization [64, 107], augmented reality [89, 99], mixed reality [14, 102] and image/video sharing service [36, 37]. The goal of the 6 degrees of freedom (DoF) camera pose estimation is to compute both 3-DoF orientation and 3-DoF location of the query image with respect to a given 3D reference scene. Over the past decades, many 6-DoF camera pose estimation approaches have been proposed. In this Section, we review the 6-DoF camera pose estimation approaches in literature.

3.1 Categorize the prior art based on the data types

If we categorize the camera pose estimation methods based on the query and reference data types, we can summarize them in 4 categories as shown in Fig.3.1.

1. Query data is a 2D image and reference data is a set of 2D images. One common approach is the image retrieval-based method. This method approximates the query image's 6-DoF pose with the most similar reference image's 6-DoF pose. Because this method is usually effective when the reference images is in a large scale [108, 120, 135] and it can be robust to changing environmental conditions [3, 136], it is widely used for the application of place recognition [3, 108, 136]. Another approach is to compute the relative camera pose between the query image and each reference image [48], and then fuse these relative camera pose into a final 6-DoF camera pose by minimizing a defined geometry error [130].
2. Query data is a 2D image and reference data is 3D point cloud and images. One popular approach [65, 122, 131] is to find the 2D-3D correspondences between the 2D query image and the 3D point cloud through matching feature descriptors [1, 6, 70, 117], and then these 2D-3D correspondences are



Query data	Reference data
Single 2D image	Multiple 2D images
Single 2D image	3D structure
Multiple 2D images	Multiple 2D images
Multiple 2D images	3D structure

Figure 3.1 Categories of camera pose estimation methods based on the different types of query and reference data. Query data is either single image or multiple images, and reference data is images, LIDAR point clouds, or both of them.

used to estimate the 6-DoF camera pose of the query image with Perspective-n-Point methods [41, 137]. Descriptor matching can be improved by efficient search [8, 21, 154], prioritization [72, 122], geometric constraints [16, 131], semantic verification [133], etc. Another popular approach [99, 107, 139] computes the 6-DoF camera pose by minimizing a cost function directly in the 6D space of camera poses without creating the 2D-3D correspondences.

3. Query data is a sequence of 2D images and the reference data is a set of 2D images. Sequence-based methods take a set of images as the query data and match it with the reference images, and they usually take image-retrieval approaches [81, 92, 97] by utilizing the temporal coherence in the data to match the query and reference sequences.

4. Query data is a sequence of 2D images and the reference data is 3D point cloud and images. One popular approach reconstructs a 3D model from the input sequence, and then match the reconsrctured 3D model with the 3D reference data [76, 153].

All of these 4 categories have been used in this PhD work. Chapter 4 belongs to 2nd category. Section 5.1 belongs to 1st category. Section 5.2 belongs to 3rd category. and Chapter 6 belongs to 4th category.

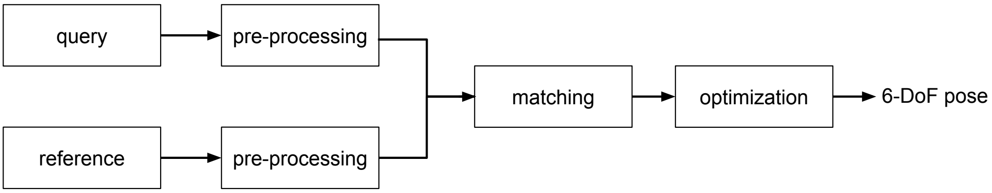


Figure 3.2 Shared components for different 6-DoF camera pose estimation approaches.

3.2 Categorize the prior art based on the method types

Based on the types of approaches, the 6-DoF camera pose estimation methods can be divided into 3 categories: (1) *indirect* approaches, (2) *direct* approaches, and (3) *hybrid* approaches. The (1) *indirect* approach establishes 2D-3D correspondences between the query image and reference point cloud to compute the camera pose, and it can be considered as a combinatorial optimization method. The (2) *direct* approach estimates query's camera pose by optimizing a cost function in 6D pose space, and the cost function can be defined by the difference of the query image with a rendered synthetic view from a reference 3D point cloud. The (3) *hybrid* approach combines both the *direct* and *indirect* methods for camera pose estimation. We present the shared essential building block among these 3 different approaches, and then follow up with the detail discussions of each approach.

Among these 3 different approaches, there are some shared essential building blocks and we summarize those shared components as shown in Fig. 3.2.

1. *Pre-processing* is a common step for both query and reference data, and the purpose is to make the query and reference data to be represented in a format that is easier for further matching. A lot of literature have been studied in the field. Some of the pre-processing methods for query data are: (1) extracting image features representation [6, 15, 18, 20, 22, 74, 116]; (2) converting the query image to illumination-invariant color spaces [83, 88]; (3) generating 3D structures from multiple images with structure-from-motion [76, 153]. Some of the pre-processing methods for reference data are: (1) extracting image features; (2) generating synthetic 2D views from 3D reference data [40, 53, 136]; (3) associating 3D reference points to visual words [121].
2. *Matching* between the query and reference is commonly applied after the *pre-*

processing steps. The matching methods are usually straightforward for the *direct* camera pose estimation methods, e.g. compare the intensity values of the query image and reference image [99, 139]. For *indirect* camera pose estimation methods, descriptor matching is utilized for the extracted features from query and reference data. Depending on the amount of the data to be matched, the exhaustive search or other more efficent search methods [8, 121, 154] can be used.

3. *Optimization* is a common approach to estimate camera pose with given *matching* results. The *direct* methods get the camera pose by optimizing a cost function in 6D pose space directly, and coarse-to-fine grid searches [40] or gradient methods [118] are usually used in the optimization process. For the *indirect* methods, given 2D-3D correspondences, Perspective-n-Point(PnP) methods [41, 137] and RANSAC [31, 137] are usually used to compute the 6-DoF camera pose, which can be viewed as combinatorial optimization methods.

The *indirect* approach can be considered as a combinatorial optimization approach, because it computes the camera pose using Perspective-n-Point (PnP) and RANSAC on 2D-3D correspondences and different combinations of the 2D-3D correspondences lead to different estimated camera pose. One popular way to find the 2D-3D correspondences is via 2D-2D feature matching between query and reference images. Those image patterns which differ from their immediate neighborhood are usually considered in 2D image feature detection, e.g. corners [91, 116], blobs [6, 58, 74], and feature detectors' performance varies in terms of invariance to rotation, scale or even deformation [138]. Then feature descriptors [1, 2, 6, 15, 18, 70, 74, 117, 134] provide robust description of a patch centered around each detected feature points. Besides the above hand-crafted feature detectors and descriptors, learned alternatives can be used to replace detector [123, 150], descriptor [125, 126], or both of them [23, 104]. The 2D-2D correspondences can be matched with exhaustive search or more efficently approximate nearest neighbor search [94], and 2D-3D correspondences are indirectly established. Finally, PnP [41, 137] and RANSAC [31, 137] take the 2D-3D correspondences and estimate the 6-DoF camera pose of the query image with regarding to the 3D reference. The idea of the *indirect* approach has been widely used in Structure-from-Motion methods [124, 127, 145]. For example, incremental Structure-from-Motion methods [124, 145] estimate the camera pose of query images one by one into the 3D model. Many Simultaneous Localization and Mapping

(SLAM) [67, 95, 96] and Visual Odometry (VO) [66, 100] algorithms also choose the *indirect* approach as one important component in their solutions. In the presence of both reference point clouds and reference images, the *indirect* approach can be utilized to estimate 6-DoF camera pose of the query image with regarding to point cloud and reference images [64].

The *direct* approach estimates 6-DoF camera pose by minimizing a cost function directly in the multidimensional space of the camera pose, and the final camera pose is optimized by either gradient [118] or grid search [40] methods. Usually, the cost function compares the query image with a reference image or a generated synthetic view from a 3D point cloud. Photometric error is one of the most popular cost functions in *direct* approach, and it is widely used in SLAM [25, 99, 139] and Visual Odometry [24, 144] methods. Compared with *indirect* methods, the *direct* methods have the possibility to use all information in the image instead of sparse local features for optimization, which leads higher accuracy and robustness in environments with little feature points [24]. By using parallel computing, direct method DTAM [99] achieves real-time camera tracking and reconstruction results with the help of GPU hardware. Another direct monocular SLAM approach, LSD-SLAM [25], can build a large-scale map and run in real-time on a CPU. However, the *direct* method are arguably more sensitive to environmental condition changes, e.g. lighting and view points [99]. To improve the robustness of the cost function, NID-SLAM [107] proposes a normalized information distance as the cost function and it outperforms a few feature-based and photometric-based approaches in two challenging data sets. However, NID-SLAM is less robust to depth errors and relies on good initialization [107].

The *hybrid* approach uses both the *direct* and the *indirect* components in the camera pose estimation, and the *hybrid* approach usually combines the success-factors (e.g. tracking features with invariant properties, key-frame selection) of the *indirect* approach with the accuracy and speed of the *direct* approach. SVO [32] is an example of *hybrid* approach, where the feature extraction is only needed when a key-frame arrives and the camera pose relative to the previous frame is computed through minimizing photometric error of the corresponding feature-patches. This SVO method [32] has increased speed performance comparing with the *indirect* approach where the feature extraction and matching is usually required for each query image. In contrast to the *direct* method, the SVO method uses many small patches instead of few large planar patches. Similarly, PL-SVO [44] extends the SVO [32] approach to work with line segments. However, both SVO [32] and PL-SVO [44] is targeting on Visual Odometry problem and working with video frames.

4 SINGLE-QUERY 3D CAMERA POSE ESTIMATION (PUBLICATION I)

4.1 Introduction

In this Chapter we focus on single-query 6-DoF camera pose estimation and it is related to Publication I. Although there are a lot of different approaches to solve the pose estimation problem in the literature as we described in Chapter 3, it was necessary to conduct a rigorous and objective comparative analysis in order to identify the strengths and weaknesses of each approach. We investigated the performance of 4 different camera pose estimation methods, i.e. feature-based *indirect* method [64], photometric-based *direct* method [139], mutual-information-based *direct* method [107], and a *hybrid* method for single-query 6-DoF pose estimation in two public datasets: KITTI [42] (see Section 2.4.2) and Oxford RobotCar [82] (see Section 2.4.1).

4.2 Methods

We present 3 methods for the single query camera pose estimation in this Section. We describe each method with the most basic case of the reference data, i.e. one *reference tuple*. One *reference tuple* consists of a street-view snapshot and a point cloud as shown in Fig. 4.1. Using multiple *reference tuples* for single-query camera pose estimation is discussed in the next Section.

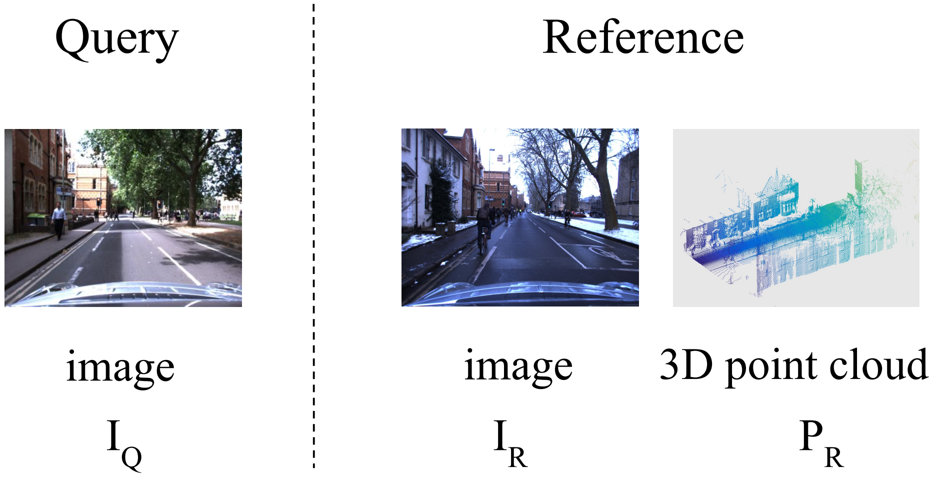


Figure 4.1 Inputs for the pose estimation methods in the simplest scenario: a query image I_Q and a *reference tuple* (I_R, P_R), where I_R is a single reference image and P_R is the registered 3D point cloud associated to I_R. Both the the point cloud P_R and the camera pose of the reference image I_R are defined in a common world coordinate system.

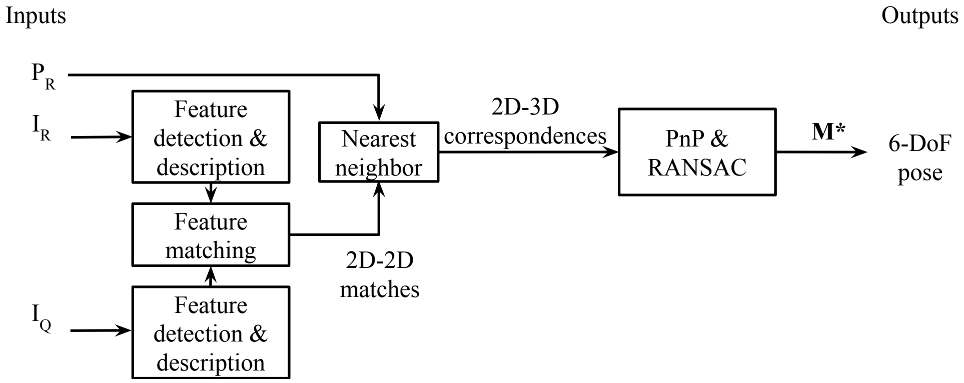


Figure 4.2 Flowchart of feature-based camera pose estimation. I_Q is the query image. The reference image I_R and the 3D point cloud P_R are pre-registered and defined in a reference coordinate system. $\mathbf{M}^*$ is the estimated extrinsic matrix for the query image.

4.2.1 The indirect method

The indirect method extracts image features from both query and reference images to indirectly estimate the camera pose. There are mainly 4 major steps, namely (1) feature detection and description, (2) feature matching, (3) 2D-3D correspondences grouping, and (4) Perspective-n-Point pose estimation. Fig. 4.2 illustrates a general flowchart of the indirect method.

Firstly, a 2D image feature detector [6] locates the image features in both the query (I_Q) and reference (I_R) images, and then a feature descriptor [6] is used to describe the detected features.

Secondly, in the feature matching stage, the exhaustive search is used to find the 2D-2D correspondences between the query image features and the reference image features. These 2D-2D correspondences set S can be presented as follows:

$$S = \{(\mathbf{p}_Q^{(1)}, \mathbf{p}_R^{(1)}), (\mathbf{p}_Q^{(2)}, \mathbf{p}_R^{(2)}), \ldots, (\mathbf{p}_Q^{(n)}, \mathbf{p}_R^{(n)})\} \tag{4.1}$$

where $\mathbf{p}_Q^{(i)} = [u_Q^{(i)}, v_Q^{(i)}]^T$ and $\mathbf{p}_R^{(i)} = [u_R^{(i)}, v_R^{(i)}]^T$ are the ith 2D feature locations on query and reference images.

Thirdly, to find the 2D-3D correspondences, the reference point cloud P_R is projected on the reference image with known camera calibration parameters. It can be expressed as equation (4.2),

$$\mathbf{p}^{(i)} = \mathbf{K}_R \mathbf{M}_R \mathbf{P}_R^{(i)} \tag{4.2}$$

where $\mathbf{P}_R^{(i)}$ is the i-th point in 3D point cloud P_R, $\mathbf{M}_R$ and $\mathbf{K}_R$ is the extrinsic matrix and the intrinsic matrix for the reference camera, and $\mathbf{p}^{(i)}$ is the i-th point in the 2D projections p. p is defined as:

$$p = \{\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \ldots, \mathbf{p}^{(n)}\} \tag{4.3}$$

Then, nearest neighbor search [33] finds the correspondences in p for reference image feature points in S. For example, the j-th reference feature point $\mathbf{p}_R^{(j)}$ in S is associated to the k-th point of the 2D projection set p with the nearest neighbor (NN) [33] by:

$$k = \mathrm{NN}(\mathbf{p}_R^{(j)}, p), \quad k \in \{1, 2 \ldots, m\} \tag{4.4}$$

Given index k, the j-th reference point can be expressed in homogeneous coordinates

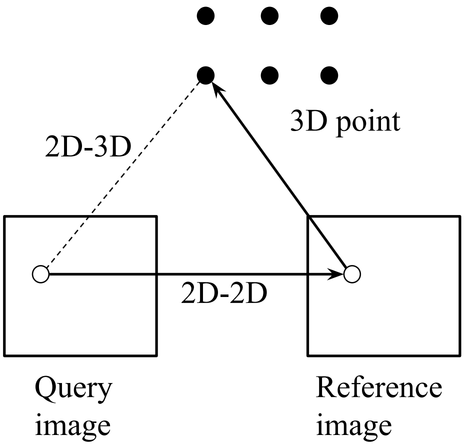


Figure 4.3 Building 2D-3D correspondences through the 2D-2D matched features and 3D points.

as follow:

$$\begin{bmatrix} \mathbf{p}_R^{(j)} z^{(k)} \\ z^{(k)} \end{bmatrix} \tag{4.5}$$

where $z^{(k)}$ is the depth of the associated k-th point in the 2D projection set p and $z^{(k)}$ is used as the estimated depth value for $\mathbf{p}_R^{(j)}$. The 3D coordinates of reference image feature points in set S can be calculated with the inverse of intrinsic $\mathbf{K}$ together with depth values, as follows:

$$\mathbf{P}^{(j)} = \mathbf{K}^{-1} \begin{bmatrix} \mathbf{p}_R^{(j)} z^{(k)} \\ z^{(k)} \end{bmatrix} \tag{4.6}$$

where $z^{(k)}$ is the estimated depth value of the reference image feature point $\mathbf{p}_R^{(j)}$, $\mathbf{K}$ is the intrinsic matrix of the reference camera, $\mathbf{p}_R^{(j)}$ is the j-th reference feature point, and $\mathbf{P}^{(j)}$ is the 3D coordinates in reference camera frame. Further utilizing the 2D-2D correspondences from equation (4.1), we indirectly find the 2D-3D correspondences set $\hat{S}$ between query image feature points and 3D points shown as Fig. 4.3 and $\hat{S}$ can be expressed as follows:

$$\hat{S} = \{(\mathbf{p}_Q^{(1)}, \mathbf{P}^{(1)}), (\mathbf{p}_Q^{(2)}, \mathbf{P}^{(2)})..., (\mathbf{p}_Q^{(n)}, \mathbf{P}^{(n)})\} \tag{4.7}$$

where $\mathbf{p}_Q^{(i)}$ is the i-th 2D feature location of the query image, and $\mathbf{P}^{(i)}$ is the corresponding i-th 3D point in the reference camera coordinate.

Finally, a Perspective-n-Point (PnP) solver [41] together with a RANSAC [137] are applied to estimate the 6-DoF camera pose of the query image in the reference coordinate system, by optimizing the re-projection error,

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} \sum_{\forall i} ||\mathbf{p}_Q^{(i)} - \mathbf{KMP}^{(i)}||, \quad i \in \{1,2\ldots,n\} \tag{4.8}$$

where $\mathbf{p}_Q^{(i)}$ is the i-th feature point at the query image, $\mathbf{K}$ is the query camera's intrinsic matrix, $\mathbf{M}$ is the sought query camera's extrinsic matrix, $\mathbf{P}^{(i)}$ is its corresponding 3D coordinate, $||\cdot||$ calculates the euclidean distance, and $\mathbf{M}^*$ is the best estimate.

4.2.2 The direct method

The direct method for camera pose estimation calculates the 6-DoF camera pose by minimizing a cost function directly in 6D space. In contrast of the indirect method, the direct method does not need to extract the 2D image features, but it needs to design a cost function and an optimizing approach to find the minimum. In this Section, we present two direct methods with different cost functions: direct photometric-based camera pose estimation and direct mutual-information-based camera pose estimation.

4.2.2.1 Direct photometric-based camera pose estimation

The direct photometric-based approaches [25, 99, 139] use the photometric values in the cost function, and the cost function is directly optimized in the 6-DoF camera pose space. The goal is to find the best estimated camera pose that minimizes the photometric cost function, and the photometric values of the query image is usually directly compared with a synthetic image rendered from the reference 3D data, similar as the following equation:

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} \mathrm{RES}(I_Q, I_S), \tag{4.9}$$

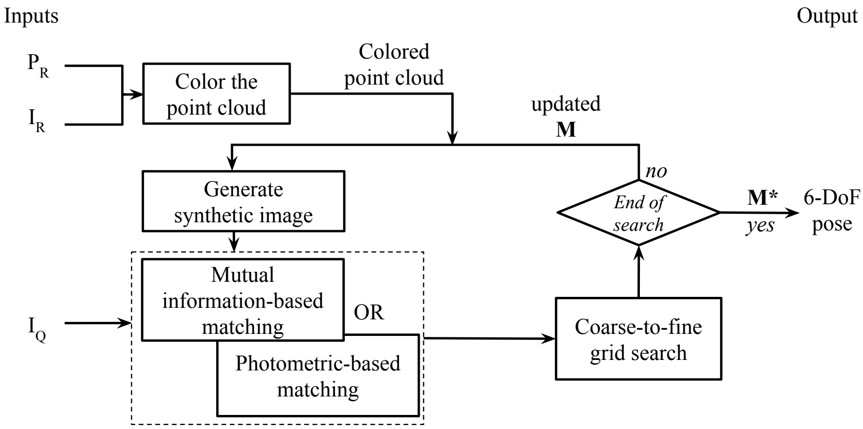


Figure 4.4 Block diagram of direct photometric-based and mutual information based camera pose estimation. I_Q is the query image. The reference image I_R and the 3D point cloud P_R are pre-registered and defined in a reference coordinate system. $\mathbf{M}^*$ is the estimated query image's extrinsic matrix.

where, the cost function RES is defined as the photometric error and it can be described as:

$$\mathrm{RES}(I_Q, I_S) = \frac{1}{\mu} \sum_{(u,v)\in I_S} (I_Q(u,v) - I_S(u,v))^2 \tag{4.10}$$

In equation (4.10) query image is I_Q, synthetic view is I_S, and μ is the number of pixels in I_S.

The block diagram of photometric-based camera pose estimation is illustrated in Fig. 4.4. Because the cost function is the only difference between the presented photometric-based and mutual-information-based methods, both methods share the same block diagram but different cost functions in the matching stage. The photometric-based method consists of three main steps: (1) synthetic image generation, (2) photometric matching, and (3) coarse-to-fine search. The methods works as follows.

Firstly, in order to generate synthetic views that can be compared with the query image as shown in equation (4.9), the 3D point cloud P_R is "colored" by projecting each 3D point on the reference image I_R and then taking the color (or intensity value) from the projection point. This process can be expressed as:

$$I(\mathbf{P}_R^{(i)}) \leftarrow f(\mathbf{p}_R^{(i)}, I_R), \quad I_R \in \mathbb{R}^2 \tag{4.11}$$

where I_R is the reference image, $\mathbf{p}_R^{(i)}$ is the 2D projection point of the i-th 3D point $\mathbf{P}_R^{(i)}$, f is the cubic interpolation function since the projection point is not necessarily landing in the middle of an image pixel, and $I(\mathbf{P}_R^{(i)})$ is the estimated intensity (or color) value of the 3D point $\mathbf{P}_R^{(i)}$. Then, we can render synthetic views I_S with a controlled view point $\mathbf{M}$ and intrinsic parameters $\mathbf{K}$,

$$I_S(\mathbf{KMP}_R^{(i)}) \leftarrow I(\mathbf{P}_R^{(i)}), \tag{4.12}$$

where $I(\mathbf{P}_R^{(i)})$ is the intensity value of the i-th 3D point $\mathbf{P}_R^{(i)}$, and $I_S(\mathbf{KMP}_R^{(i)})$ is the intensity value of the rendered synthetic view.

Secondly, the rendered synthetic view is compared with the query image as equation (4.9). To enhance the robustness of the cost function (4.10), a Gaussian filter is used to smooth the query image and an M-estimator [52] is designed to remove some outliers. The main idea of the designed M-estimator is to give smaller weights for residual values which might be considered as outliers through analyzing the residual distribution. Therefore, the cost function (4.10) can be rewritten as:

$$\mathrm{RES}(I_Q, I_S) = \frac{1}{\lambda} \sum_{\forall(u,v)} (\mathrm{Diff}(u,v))^2 w(u,v) \tag{4.13}$$

where $\mathrm{Diff}(u,v)$ and weights $w(u,v)$ are defined as (4.14) and (4.15) respectively, and λ is the number of nonzero weights.

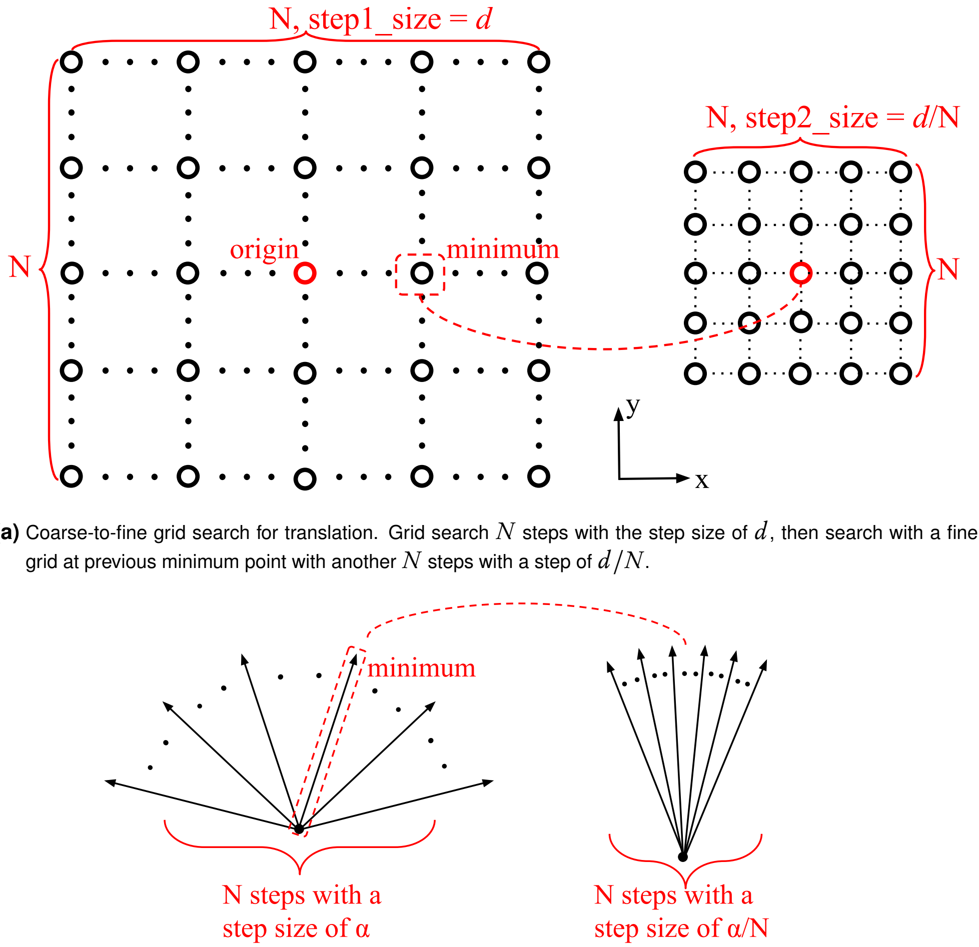
$$\mathrm{Diff}(u,v) = (I_Q(u,v) - I'_S(u,v))^2, (u,v) \in I'_S \tag{4.14}$$

where I_Q is the query image and I'_S is the low-pass filtered synthetic image.

$$w(u,v) = \begin{cases} 0, & \text{if } \mathrm{Diff}(u,v) > \theta \\ 1, & \text{otherwise} \end{cases} \tag{4.15}$$

where θ is the median value of the vector $\mathrm{Diff}(u,v)$ and (u,v) is the pixel coordinates in I'_S.

Thirdly, a coarse-to-fine grid search is used to find the best estimated transform $\mathbf{M}^*$ in equation (4.9). The change of the controlled camera view point $\mathbf{M}$ leads to difference residual values in equation (4.13), and we aim to find the $\mathbf{M}^*$ by trying different combinations of camera rotation and translation. The main idea of the coarse-



(a) Coarse-to-fine grid search for translation. Grid search N steps with the step size of d, then search with a finer grid at previous minimum point with another N steps with a step of d/N.

(b) Coarse-to-fine grid search for orientation. Grid search N steps with the angular step of α, then refine the search by another N steps with a step of α/N.

Figure 4.5 Two-step coarse to fine search.

to-fine grid search is to take a relative bigger step in the 1st round of grid search, then followed by a smaller step in the next round of grid search. Two examples of a 2-step coarse-to-find grid search is illustrated in Fig. 4.5.

Finally, the 6-DoF estimated camera pose is obtained from the best estimated transformation $\mathbf{M}^*$. It should be noted that in common tracking applications where transformation baseline is small, fast optimization can be implemented by using gradient-based optimization [139].

4.2.2.2 Direct mutual-information-based camera pose estimation

Mutual information [142] is the measure of the mutual dependence between two variables and it is widely used in medical image registration over different modalities [84]. Compared with photometric-based approach, Pascoe et al. [107] shows mutual information provides robustness to appearance, weather and structural changes for camera pose estimation. The direct mutual-information-based approach use normalized mutual information to design the cost function. As Fig. 4.2 illustrated, the cost function for matching the query image and synthetic images is main difference between photometric-based and mutual information-based methods, so the designed the cost function is discussed below.

With normalized mutual information (NMI) [87], the cost function and the optimization task can be defined as follows:

$$\mathbf{M}^* = \arg\min_{\mathbf{M}}(1 - NMI(I_Q, I_S)), \tag{4.16}$$

where I_Q and I_S are the query and synthetic images respectively, $\mathbf{M}$ is the controlled camera extrinsic matrix, $\mathbf{M}^*$ is the best estimate, and NMI is designed as:

$$NMI(I_S, I_Q) = \frac{MI(I_S, I_Q)}{max(H(I_S), H(I_Q))} \tag{4.17}$$

and mutual information (MI) is defined as:

$$MI(I_S, I_Q) = H(I_S) + H(I_Q) - H(I_S, I_Q) \ , \tag{4.18}$$

where $H(I_S)$ and $H(I_Q)$ are the marginal entropies of I_S and I_Q, and $H(I_S, I_Q)$ is the joint entropy of I_S and I_Q.

4.2.3 The hybrid method

The hybrid approach for camera pose estimation takes the strength of both indirect feature-based pose estimation and direct mutual-information-based pose estimation. Fig. 4.6 shows the block diagram of the hybrid method, and specific steps works as follows.

In the hybrid method, a feature detector and a feature descriptor are used to ex-

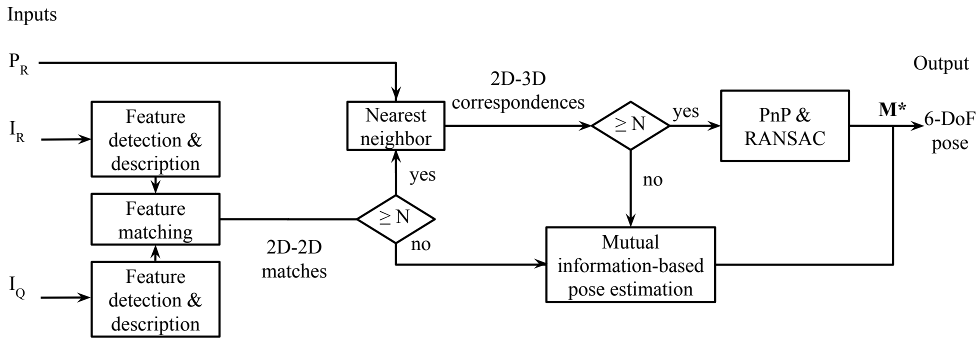


Figure 4.6 Hybrid approach of camera pose estimation. I_Q is the query image, I_R is the reference image, and P_R is the 3D point cloud P_R which is pre-registered with P_R. N is the minimum number of required pairs, and 4 is used in our experiments.

tract 2D features from both the query I_Q and reference image I_R. After the feature matching, if the number of matched features is less than the threshold N, the mutual information-based approach described in Section 4.2.2.2 will be used to compute the camera pose. Otherwise, continue with feature-based approach to compute the 2D-3D correspondences. If the number of 2D-3D correspondences is less than the threshold N, mutual information-based approach is adopted. Otherwise, a PnP solver [41] together with RANSAC [137] are used to compute the camera pose.

4.3 Comparative methodology

In order to identify the advantages and disadvantages of each method, it was necessary to consider different scenarios. Therefore, we designed three comprehensive sets of experiments for the single-query 6-DoF camera pose estimation, and these experiments allow us to systematically evaluate the performance of each selected methods in KITTI [42] and Oxford RobotCar [82] dataset. Three sets of experiments are described as follow:

1. Single-reference camera pose estimation
   We evaluate the performance of the selected methods with the most basic case of reference data, where there is only one *reference tuple*. One *reference tuple* is defined as one reference image together with its corresponding point cloud, as

shown in Fig. 4.1. Given a query image, a *reference tuple* is randomly selected in a region that centers at the actual location of the query image with a radius r. The reason of using random selection is to analyze how the difference in overlap between query and reference image effect the performance of the selected methods. The potential overlap between the query and reference image decrease with the increase of the radius r. For both the photometric-based and the mutual-information-based methods, the random selection of the *reference tuple* can be viewed as initialization with different errors. The estimated camera pose is evaluated in terms of translation and rotation error comparing with the ground truth.

2. Multiple-reference camera pose estimation
   We study the benefits of involving multiple reference images in the camera pose estimation. Each method is provided with one query image and *k reference tuples*, which contain *k* reference images together with their corresponding point cloud. Each *reference tuple* is selected and used the same way as the single-reference camera pose estimation. To fuse the multiple camera poses estimated from these *reference tuples*, four fusing methods were evaluated, namely (1) *max number of matched features*. It only selects the *reference tuple* that has the max number of matched features with the query image. (2) *Simple average*. It takes the average of the estimated candidate camera poses as the final camera pose. (3) *Weighted average*. It uses the number of the matched features between query and reference image as the weights to average the camera pose candidates. (4) *Robust weighted average*. It finds the maximum number of matches K between query and references images, and only uses the reference images with at least $K/2$ matched features. Then it computes weighted average among the candidate camera poses. Finally, the fused camera pose is considered as the final estimation and assessed with the ground truth.

3. Camera pose estimation with large uncertainties
   We investigate the camera pose estimation performance in the case of large uncertainties for the query data. The uncertainties can be represented as the search radius r for the *reference tuple*, and the larger the search radius leads to larger uncertainty. Random selection is no longer practical because there is low chance that the randomly selected reference images has any overlap compared with the query image. Therefore, the camera pose estimation perform

localization in a hierarchical fashion using image retrieval [108] as an initial step to get several *reference tuples* and then followed with selected camera pose estimation methods. Finally, the *robust weighted average* is used to compute the final camera pose by fusing the camera pose candidates generated from multiple *reference tuples*.

4.4 Experiments

Two public datasets were used in our experiments, including 11 different sequences from KITTI dataset [42] and 5 traversals with the same rout but completely different environmental conditions from Oxford RobotCar dataset [82]. The details of the used sequences from KITTI and Oxford RobotCar datasets were discussed in Section 2.4.2 and 2.4.1. The experiments with KITTI dataset were designed to evaluate each method's performance in ideal conditions (e.g. same illumination and weather conditions), and each sequence was processed independently. In a sequence, randomly selected 10% of the total images were used as query images, and the rest of the 90% images were treated as the potential reference images. The experiments with Oxford RobotCar dataset focused on the experimental setting of query data and reference data were from completely different environmental conditions. The selected 5 traversals from Oxford RobotCar dataset were respectively captured in a overcast, sunny, night, rainy, or snowy day. In each query sequence, randomly selected 10% of all the images were used as query images. The main observations from each experiments were summarized below.

1. Single-reference camera pose estimation
   We find that the feature-based approach is more accurate in pose estimation as long as it can find 4 consistent feature matches in both ideal environment conditions (KITTI dataset) and realistic environment conditions (Oxford RobotCar dataset) with random reference image selection. Both the photometric and mutual-information-based approaches are sensitive to bad initialization, but the mutual-information-based approach is more robust than feature-based approach in terms of the success rate under different environmental conditions. When analyzing the same pose estimation method for different uncertainty radii, the success rates of all approaches decrease with the increase of the uncertainty radius.

2. Multiple-reference camera pose estimation
   The increase of the number of reference images improves the success rate of each methods. Feature-based approach has the highest success rate among different approaches in the KITTI dataset, but has the lowest success rate in the Oxford RobotCar dataset. However, the mutual-information-based approach has the highest success rate in Oxford RobotCar dataset. In other words, mutual information is more robust than the two other approaches under changing environmental conditions. This finding is consistent with our results in the single reference scenario. The *robust weighted average* method is a light approach and can be easily adapted by all the tested estimation methods producing good results.

3. Camera pose estimation with large uncertainties
   The mutual-information-based approach is more robust than the feature-based or photometric-based approaches, which is consistent with the findings in the single reference and multi-reference scenarios. The *hybrid* approach outperforms all other approaches in terms of success rate when the query and reference images have very different imaging conditions. This confirms that the hybrid method leverages complementary properties of the feature- based and mutual-information-based methods.

4.5 Summary

We conducted comprehensive experiments to investigate the performance of *indirect*, *direct* and *hybrid* single-query 6-DoF camera pose estimation approaches in two public benchmarking datasets. Our experiments showed when there are at least 4 consistent feature points between query and reference images, the feature-based approach produced more precious results than both photometric-based and mutual-information based approaches. Mutual-information-based and photometric-based approaches were more sensitive to initialization than feature-based approach, however, mutual-information-based method was more robust to environmental changes compared to feature-based and photometric-based approaches. The hybrid approach exploited the strength from both the direct and indirect components and the its performance was on par or superior to other approaches.

5 APPLICATIONS OF 3D CAMERA POSE

In this Chapter, we propose two innovative augmented reality applications by using camera pose and 3D map data. The first one is a 3D map augmented photo gallery application, and the second one is an interactive video playback application.

5.1 3D map augmented mobile photo gallery (Publication II)

This Section is related to the Publication II, the granted U.S. Patent No. 9699375 and U.S. Patent No. 10102675.

5.1.1 Introduction

Camera phones are ubiquitous these days, taking a photo and sharing it with friends become one of the daily activities for the camera phone users. Among different image sharing applications, location-based image sharing functionality is a simple but widely used application. For example, a user could tag the image with the mobile phone's GPS while sharing the image. As a result, a rough location of the image is tagged, but the image sharing experience is still limited by only knowing the rough location of this image. If 6-DoF camera pose of the photo is estimated, how the image sharing and browsing applications can be further improved by using both the camera pose and the 3D map?

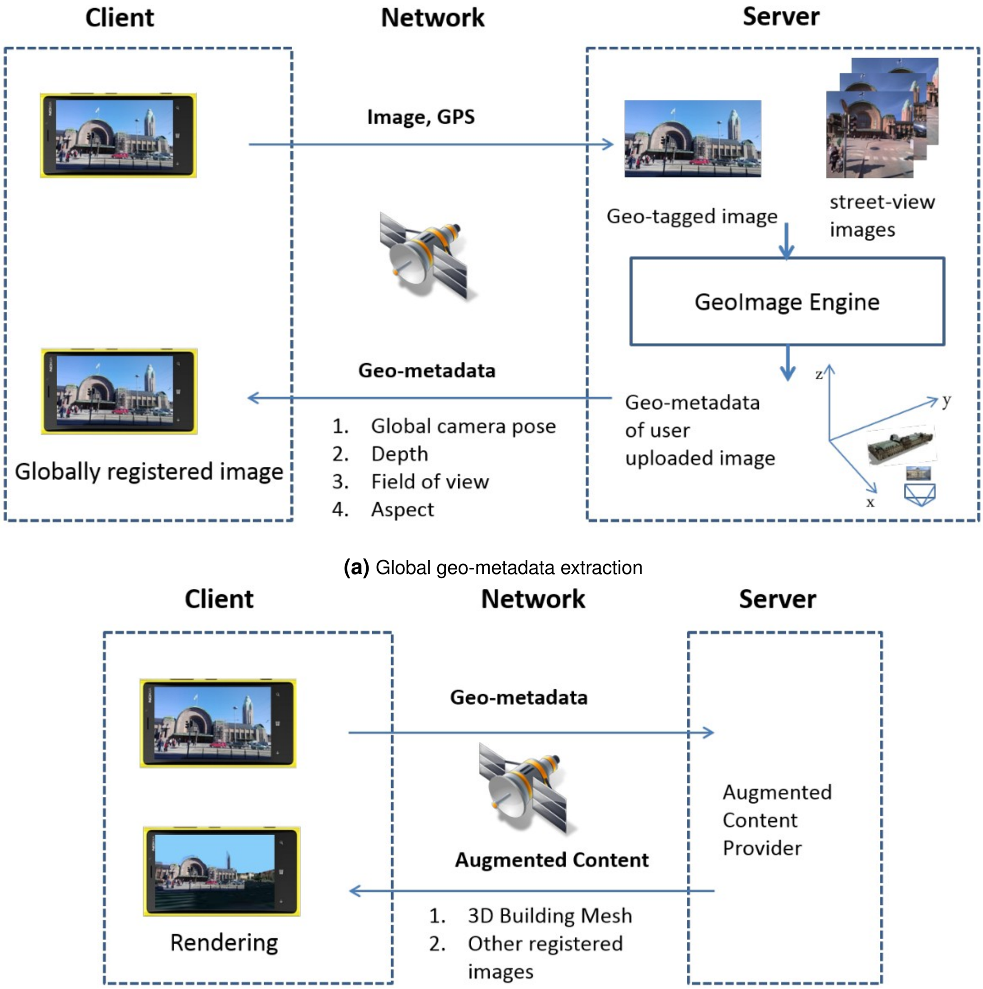
In contrast to purely GPS-based image sharing applications, some applications utilize multiple user uploaded images and their estimated camera pose to enhance the image sharing and browsing experience. Photo tourism [127] takes as input large collections of images from either personal or internet photo collections, and computes each photo's camera pose as well as a sparse 3D model of the scene. Therefore, the photo explorer interface enables the viewer to interactively move around the 3D space by transitioning between photographs. Based on photo tourism [127], Microsoft Research released Photosynth [128] which allows users to upload their images and generate their own 3D models. Then, further launched Microsoft Pix [90] helps users to create photos that take in more of the perspective or scene you are standing in front of. It allows users to freely pan and capture from side to side, up and down, back and forth, and even go back to the start to include any parts of the scene that may have missed. Similarly, a recent work from Google Research, named Neural Radiance Fields for Unconstrained Photo Collections [85], synthesizes novel views of complex outdoor scenes using unstructured collections photographs. While impressive rendering effects and good scalability have been demonstrated, these applications rely on multiple user uploaded images for creating the 3D model, and calculated the camera poses is limited in a local coordinate system instead of a global coordinate system. On the other hand, some applications use the global camera pose of the user uploaded images. Google Map enhances street-view navigation with user captured images, by allowing users to view floating thumbnails, and once a thumbnail is selected, e.g. by mouse clicking, users can change the viewing angle from the street-view image to the 2D images [46]. While it provides an interactive experience, this service is more gear to the augmentation of the street-view navigation.

We propose a novel photo gallery application [1] that provides interactive and 3D map augmented image browsing experiences, by using automatically estimated global camera pose and rendering effects in a 3D map. Using this mobile photo gallery, users can not only see the captured image during image browsing, but also expend their field of view to the surroundings by seamlessly transition from 2D image space to the 3D map space. For example, user could even see the scenes which are not initially captured in the image, i.e. left, right or even the opposite direction of the image's original viewpoint.

5.1.2 Methodology

With the consideration of a mobile application, the mobile phones usually have limited computation power and storage space. So we design a client-server architecture: the client (mobile phone) captures the images, and then in the server the global camera pose estimation of the user uploaded images are computed by exploit-

[1] 3D map augmented photo gallery: `https://junshengfu.github.io/videos/3D_photo_Album/3DPhotoAlbum.mp4`



(a) Global geo-metadata extraction

(b) Rendering in the mobile phone

Figure 5.1 Overview of the system architecture.

ing the street-view snapshots together and their global ground truth poses. For the purpose of rendering, we take mobile phone GPU computing and make real-time rendering in the client side. There are 2 main steps in the system architecture: *global geo-metadata extraction* and *mobile rendering*. The Fig. 5.1 shows the overview of the system architecture and each main step is described as follows.

5.1.2.1 Global geo-metadata extraction

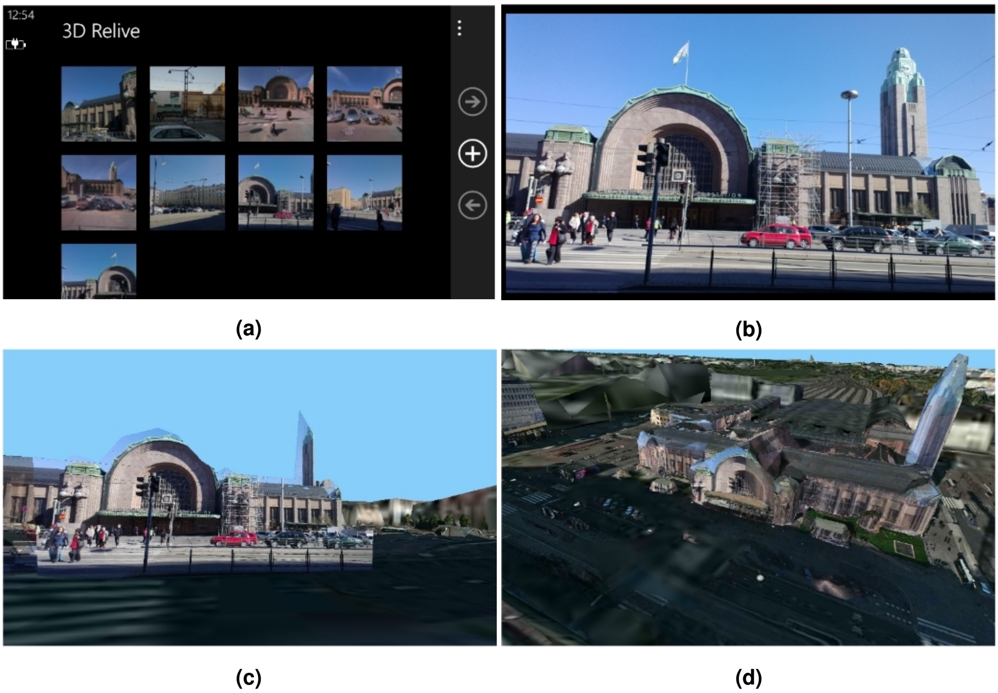
The main functionality of the module shown in Fig. 5.1a is to extract the global geo-metadata of the given image. Geo-metadata consists of query image's global camera pose, depth range, field of view, and image aspect. The reason for choosing these types of geo-metadata is that they are all required in the client rendering module. The module works as follows: firstly, client uploads a mobile image to the server together with a GPS signal. With the development of the exchangeable image file format (Exif), many cameras and mobile phones have a built-in GPS receiver that stores the GPS information in the Exif header when a picture is taken. We name this kind of images as "geo-tagged" images. Secondly, given a "geo-tagged" image, GeoImage Engine takes the its GPS and starts to search for n closest reference street-view images in the server database (n is 200 in the reported results, and this parameter can be adjusted. In our experiments our mobile phones have a built-in GPS receiver, but in case of no GPS, we could consider using the image retrieval methods [54, 108, 112, 130] to find the n most similar reference images based on the query image. Thirdly, GeoImage Engine applies feature extraction and matching among the query image and the reference images, and then sorts the reference image based on their similarities to the query image. Top k reference images are saved for further processing. Fourthly, GeoImage Engine computes the global camera pose of the query image. There are at least two different methods based on the availability of associated point cloud for the reference image. (1) In case of the presence of the associated point cloud for each reference image, the reference data are similar as the example in Fig. 4.1, so we use the camera pose estimation method discussed in Section 4.2.1. (2) In case of no point cloud, we send both the query image and the reference images into a structure-from-motion system [145]. Then, we get the camera pose of each images together with a 3D reconstruction of the scene, and everything is defined in a local coordinate system. In the experiment of this Section, we used method (2) without using point cloud. Fifthly, to represent everything in a global coordinate system, we compute a similarity transform [57] by utilizing the estimated camera locations and the known ground truth camera locations of the reference images. Finally, we apply the similarity transform to the local camera pose, and return the global metadata to the mobile client.

5.1.2.2 Mobile rendering

This Session is about rendering the augmented content in the mobile phone, and Fig. 5.1b shows the main procedures for mobile rendering. A mobile client requests nearby augmented content from the server based on the geo-metadata which were computed in the previous step. The augmented content in our experiments consists of building mesh, and other globally registered images from the content providers. The query image's geo-metadata and related augmented content from the server are all in the same global coordinate system, which enables rendering everything within a unified coordinate system. Our rendering algorithm is based on the work of view-dependent texturing [19], summarized as follows. Firstly, all rendered contents are transformed to the camera coordinate. Secondly, texture matrix is computed for each image, and the projected texture coordinate are estimated by the given camera pose and projection parameters. Thirdly, the texture matrix is passed to a pixel shader which computes a per-pixel blending factor based on the angles between the original image ray and the current viewing ray, as well as the image ray and the normal vector of the surface being projected onto. Finally, The resulting RGBA pixels are blended with the underlying map texture or other projected images.

As a result, the photo sharing and browsing application could be more intriguing and interactive for the users. We go through the main screen-shots of this application as follows: when a user opens the photo gallery, a collection of images will be shown as in Fig. 5.2a. Then user can start browsing and select one of the images, as shown in Fig. 5.2b. If there is available augmented content for the selected image, users will see a small swing motion (see demo video[2]). Once the user pinches in the image on the mobile screen, the 2D image becomes a "door" to the 3D map world and user would "enter" the 3D map through this 2D image by a seamless transition, which is illustrated as in Fig. 5.2c. Furthermore, the user can arbitrarily change the viewing angles by navigation. For example, by clicking on the sky, user could seamlessly transit from the current viewing angle to a bird-view, and Fig. 5.2d shows a bird-view of the captured scene in the query image.

[2] `https://junshengfu.github.io/videos/3D_photo_Album/3DPhotoAlbum.mp4`



(a)

(b)

(c)

(d)

Figure 5.2 Four screen shoots from our 3D map augmented photo gallery application. **(a)** Start main screen. **(b)** Select an image in photo gallery. **(c)** Switch from 2D image view to 3D map view when user pinches in the image, and the image is projected to the build mesh. **(d)** User is able to change the viewing angle arbitrarily, and currently, user is looking at a bird view.

5.1.3 Experiments

Two sets of experiments were carried out to evaluate the performance of the proposed mobile phone application: (1) An experiment of user captured mobile images without ground truth camera pose. (2) An experiment of randomly selected street-view snapshots with ground truth camera pose. In the experiments, we used street-view snapshots as reference data in the server, and no LIDAR data was used. In the experiments, a desktop computer with the processor of Intel Core i7 CPU 3.4GHz and the memory of 16 GB was used as the server to perform camera pose estimation.

1. Experiments with camera phone-captured images
   In the first experiment, 147 images were captured by 2 users with 2 camera phones, i.e. Nokia Lumia 820 and Lumia 925, in Helsinki downtown area. The client application automatically uploaded the images to the server for pro-

cessing. Since the user generated 147 images had no available ground truth camera pose, we took qualitative measurements for the analysis of results. Experiments showed that about 35% of the mobile images were able to be successfully recovered. Failure case were mainly caused by the lack of reliable features in texture-less regions e.g. skies or ground. The average processing time was 4.7 minutes, including uploading image, processing in the server and return the metadata to the client. The speed can be improved by optimization, but the speed was sufficient for our targeted application, because the action of browsing 3D augmented contents is usually not immediately followed after the photo taking action. This is usually the case when users want to show the photo to a friend when they meet next time, or share the photo on social media.

2. Experiments with stree-view snapshots
   In the second experiment, we took 305 randomly selected street-view snapshots in Helsinki downtown area as query images and example street-view snapshots were listed in Session 2.4.3. We evaluated the accuracy of the registered camera poses by comparing the estimation results with their ground truth. The orientation error metric was the maximum error between estimated Euler angles of the camera and the ground truth Euler angles. The location error metric was the Euclidean distance between the estimated camera location and the ground truth. Experiments show that the GeoImage Engine produced very good estimates for the orientation, and the maximum orientation error among all the images was less than 0.18 degrees, and gave satisfactory estimate for positions, where 93.4% of the street-view images had translation errors that were less than 6 meters.

5.1.4 Summary

The presented 3D map augmented photo gallery application shows that utilizing the estimated camera pose together with the 3D Map data could dramatically enrich the ordinary photo sharing and browsing applications. This mobile application has a client-server architecture, and automatically computes the global camera pose of the user captured images. With the mobile rendering, users could seamlessly transit their viewing points between 2D images space and 3D map space, easily navigate in the

3D map, and travel to the nearby scenes which are not visible in the original image. Since the street-view snapshots are used as the reference data in this application, the current application is limited to images that are taken outdoor and have street-view coverage.The experimental results show good accuracy for global camera pose estimation. Without optimization, the processing time is acceptable for our targeted user cases; however, there is a room for optimization if the target user cases require faster processing time.

5.2 Augmented and interactive video playback (Publication III)

This Section is related to the Publication III, the granted U.S. Patent No. 9558559 and U.S. Patent No. 9596404.

5.2.1 Introduction

Recording a video clip becomes convenient and effortless these days, since we almost always have our camera phone in our pocket. With the emergence of the TikTok, Instagram, YouTube, Twitter, etc, it is common to take a video and post it on social media to share with your family or friends. From the Section 5.1, we know the estimated global camera pose can greatly enrich the image sharing and browsing experience, but how the estimated 6-DoF camera pose of the video frames could potentially improve the video sharing and playback experience together with a 3D map?

The process of adding geographical metadata into video, image, or other media is called *Geo-tagging*, and there is a well developed standard for saving location information in photos and video, named Exif. Usually, the Exif metadata contains a single location information for the starting point of the video. Many Mobile photo galleries applications could group videos based on their location saved in Exif, and display them on a map-view [79]. To the best of our knowledge, geo-tagging enhanced video playback systems have not been fully investigated. In contrast to applications using the GPS of the first video frame, RouteShoot [75] is a mobile app that saves the full GPS trace while video recording. During the video playback, users could not only see the video clips but also simultaneously the GPS trajectory in a map view. Similarly, many vehicles have dash cams with a GPS tracker, which allows video playback with GPS trajectory visulization [106]. However, these applications are more limited to using the camera's 2D location information rather than 6D camera pose. Facebook's 6-DoF video camera [109] is a 6-DoF 360° video capture and playback system. It has 16 cameras integrated in the system, and it supports a fully spherical capture as well as interactive video playback based on the head motions. This type of system utilizes camera pose of each integrated camera to reconstruct the surrounding, but it is much more expensive than an ordinary mobile camera phone and it is usually used for Virtual Reality applications.

We propose a geo-metadata enhanced video playback application with an ordinary mobile camera. It can provide an interactive video experience and enable video playback with augmented reality contents. Our application shows that the ordinary video sharing and playback experience could be greatly enriched by leveraging geo-metadata associated with video frames, and a video demonstration is available [3] and main screen-shots are shown in Fig. 5.3.

5.2.2 Methodology

A client-server architecture is used for our proposed augmented and interactive video playback application, and there are 3 main components in the architecture, namely, mobile client, GeoVideoEngine, and AR content server. The overview of the approach and system architecture is shown in Fig. 5.4, and the 2 main steps of the system are discussed as below.

5.2.2.1 Extraction of GeoVideo metadata

After capturing a video, the mobile client uploads this video sequence to the server, named GeoVideo Engine. Then the GeoVideo Engine computes the geo-metadata of the video, and returns back to the client. The procedure is similar as the first step in *3D map augmented photo gallery application* at Section 5.1.2.1, but the main difference is that GeoVideo Engines process a set of video key frames instead of only one query image from the camera.

[3] Augmented and interactive video playback: `https://junshengfu.github.io/videos/video_playback/videoPlayBack.mp4`

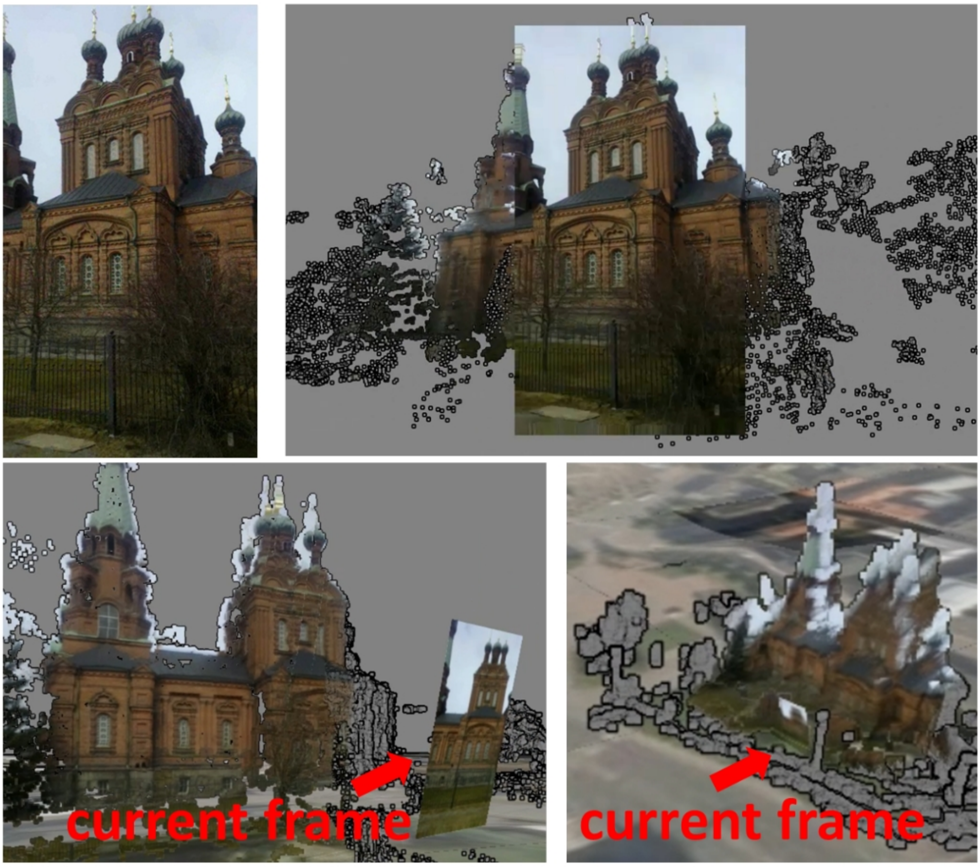


Figure 5.3 A few different video playback options are illustrated. Ordinary video playback mode (upper left): video is played back as same as the captured video; Expanded view mode (upper right): user can expend their field of view to surrounding environments that are not visible in the original video frame by the augmented 3D model; Motion trajectory(bottom left): with the scene fixed and see camera's trajectory along the scene. Bird-view (bottom right): users can arbitrarily change their viewing angles, and see the scene in an aligned and augmented map.

Once the video is sent to the server, the GeoVideo Engines automatically starts to extract the key frames of the video, and in our experiments we extract every 30th frame from the video. Then based on the GPS information of the first frame, we search for *n* closet street-view images. There are two important modules involved in GeoVideo Engine: *3D reconstruction* and *global alignment*. In the *3D reconstruction* module, structure-from-motion techniques [145] are used to reconstruct 3D point clouds and relative camera pose of both video key frames and reference street-view

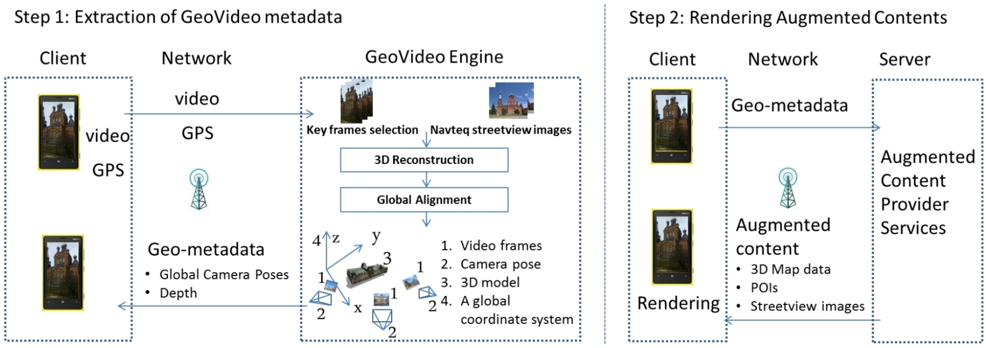


Figure 5.4 The architecture of the augmented and interactive video playback system.

images. The details of the applied structure-from-motion techniques can be found at [145, 146]. In the *global alignment* module, we aim to define the camera pose and the 3D reconstruction in a unified global coordinate system, such as Earth-Centered Earth-Fixed (ECEF) system. Since the involved street view images have camera location definitions in both local (3D reconstruction) and global coordinate systems (ground truth provided in WGS86 coordinate system), we computed a similarity transform which can convert between two coordinates [57]. By applying the estimated similarity transform, 3D reconstructed point cloud and the camera pose of the images could be further represented in a unified global coordinate system. Consequently, the global geo-metadata of the uploaded video is ready to be returned to the client.

5.2.2.2 Rendering augmented contents

In this step, client requests nearby augmented contents from the server based on the estimated geo-metadata from the video. The AR content server returns 3D models, street-view images, and point-of-interest data. All the augmented contents and video frames are transformed into the camera coordinate system, and then it is straightforward to render everything within a unified global coordinate system. The rendering algorithm is based on the work of view-dependent texturing [19], and detailed description can be seen in Section 5.1.2.2. As a result, several new and intriguing video playback options become possible:

1. Expand your field-of-view: as shown in Fig. 5.3 up-right image, the middle

part of the image is the original captured content, but users are able to expand the field-of-view by augmenting the reconstructed 3D model.

2. Video motion path visualization: as shown in Fig. 5.3 bottom-left image, we fix the 3D reconstructed scene and playback the video based on the camera pose trajectory. As a results, users can highlight the camera motion and see how the camera travels along the 3D scene.
3. Arbitrary change of viewing angles: as shown in Fig. 5.3 bottom-right, users could arbitrary change the viewing angle and it is possible to see the surrounding environment with the aligned and augmented map content.

5.2.3 Summary

We proposed and implemented a client-server architecture that uses geo-metadata to enrich the video playback experience. In contrast to existing video playback applications, our application computes the global 6-DoF camera pose of the video frames, and allow users to expend the field of view, see the 6-DoF camera trajectory, and arbitrarily change of the viewing angle. Due to the limitation of the street-view data, our application are mainly targeted on outdoor use cases. Also, video clips travel up to a few kilometers may bring difficulties to the current system, because ordinary mobile videos have only a single GPS tagging which is the GPS of the first frame, and it is used for searching nearby street-view images. Therefore the search range will need to be tuned accordingly. To overcome the challenge for long sequences, we could either turn on the GPS logging for all key frames during video capturing or use image retrieval method for finding suitable street-view images. Then, we could take the similar approaches to compute the global camera pose and rendering the video during playback.

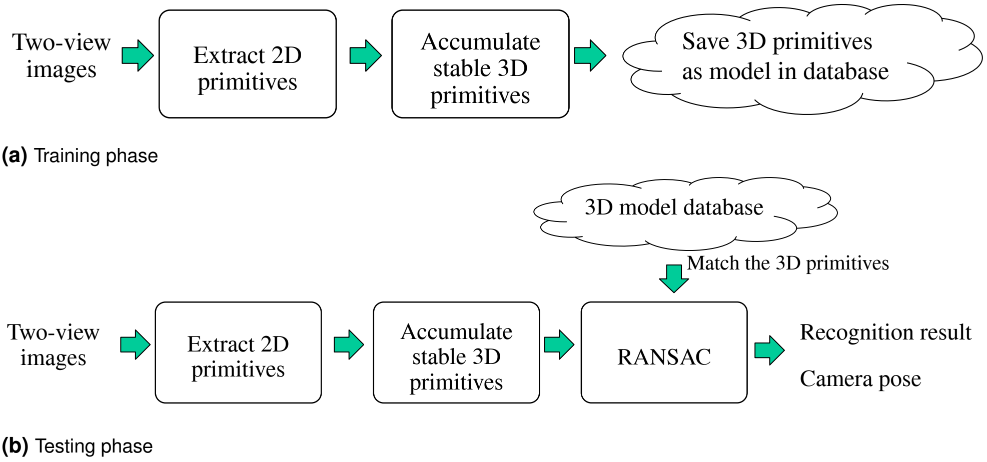
6 APPLICATION OF 3D SCENE CAPTURE: 3D PRIMITIVES BASED OBJECT AND SCENE RECOGNITION (PUBLICATION IV)

6.1 Introduction

The object and scene recognition are fundamental tasks in computer vision, and they are widely used in humanoid robots [13], autonomous vehicle [78], and augmented reality [63]. In the last decades, many different kinds of approaches have been developed and remarkable progresses have been made on object and scene recognition [5, 30, 49, 51, 148, 152]. Xie et al. [149] summarize scene recognition algorithms into 6 categories, i.e. global attribute descriptors [103], patch feature encoding [69], spatial layout pattern learning [55], discriminative region detection [71], object correlation analysis [147] and hybrid deep models [73, 111]. This work is done before the deep learning era, and we focus on patch feature encoding approaches. This work is related to Publication IV.

With the emergence of practical 3D sensors [50, 151] and the increasing popularity of multi-camera mobile phones, 3D recognition methods [5, 43, 148] are developed for object and scene recognition problems. The 3D recognition methods often use global or local feature descriptors to extract the shape of the objects or geometry of the scenes. However, they usually focus on extracting reliable 3D geometry [5, 43, 115] instead of fully exploiting visual appearance, such as color and texture, of the objects or the scenes. Many 2D recognition methods [17, 108] show that 2D visual appearance can be effective and useful for object and scene recognition. Therefore, we propose a 3D visual primitive based recognition method that exploits both the 2D visual appearances and 3D structures from multi-view images.

Our proposed 3D recognition method consists of both training and testing phases,



(a) Training phase

(b) Testing phase

Figure 6.1 Overview the proposed 3D recognition method.

and the main ideas are described in Fig. 6.1. The inputs in both training and testing phases are two-view images, and they can be a pair of two-view images or simply two images with known relative camera pose. In the training stage, given a pair of two-view images, we extract each image's 2D primitives [59] and then accumulate the stable 3D primitives to be saved as a model in the database. In the testing phase, a new pair of two-view images follow the same process in training stage to obtain the stable 3D primitives, then we apply a rand sampling-based matching to find the most similar model and give the recognition results. Also, it is interesting to notice that a rough camera pose is obtained as a side result from the matching.

6.2 Methodology

Our proposed 3D recognition method for object and scene recognition has two key components in the process, namely *construct 3D visual primitives* and *random sample-based matching*. They are further discussed in the following two sub-sections.

6.2.1 Construct 3D visual primitives

The term visual primitives derives from the primitives found in various layers of the "deep vision hierarchy" [68], and they are essentially sparse image descriptors existing both in 2D image space and 3D space. 2D visual primitives present condensed representation of the 2D image, and 2D visual primitives can be mainly grouped into 4 categories, namely constant color region, edge, junction, and texture [68]. The edge primitives are selected to be used for our object and scene recognition due to their proven effectiveness in representation of the object and scene [110]. To compute the 2D visual primitives from the two-view images, we use a regular spatial grid where circular patches are extracted by quadrature filters [29] and number of extracted 2D visual primitives can be adjusted by tuning the filtering parameters. Then, 2D visual primitives are categorized based on the computational intrinsic dimensionality [59]. The extracted 2D visual primitives can be expressed as

$$\pi = (x, \theta, \phi, \mathbf{c}) \tag{6.1}$$

where x is the 2D image position, θ is the local orientation angle of an edge/line, ϕ is the local phase of an edge/line, and $\mathbf{c}$ is a vector of RGB values.

3D visual primitives are accumulated from 2D visual primitives of two images with a bit different viewing angles. Each 2D visual primitive from one input image will be matched with every primitives from the other image based on (6.1). The requirements of the putative matches are as follows: (1) their color, orientation and phase must match; (2) the position of the primitives must lie on their corresponding epipolar lines; (3) the distance of the matches should not be greater than 1.5 times the patch size. The accumulated 3D visual primitives are encoded as

$$\Pi = (X, \mathbf{n}, \Theta, \Phi, C) \tag{6.2}$$

where X is the 3D location of the 3D visual primitive, $\mathbf{n}$ is the surface normal, Θ the edge/line orientation, Φ the edge/line phase and C the color vector constructed by the weighted average of the corresponding color values from input images. Fig. 6.2 shows an example of constructing 3D visual primitives from two extracted 2D visual primitives.

6.2.2 Random sampling-based 3D matching

Random Sample Consensus (RANSAC) is an iterative method introduced by Fischler and Bolles [31] for fitting a model to experimental data. Our designed RANSAC is similar to the one used by Papazov and Burschka [105], but their experimental data

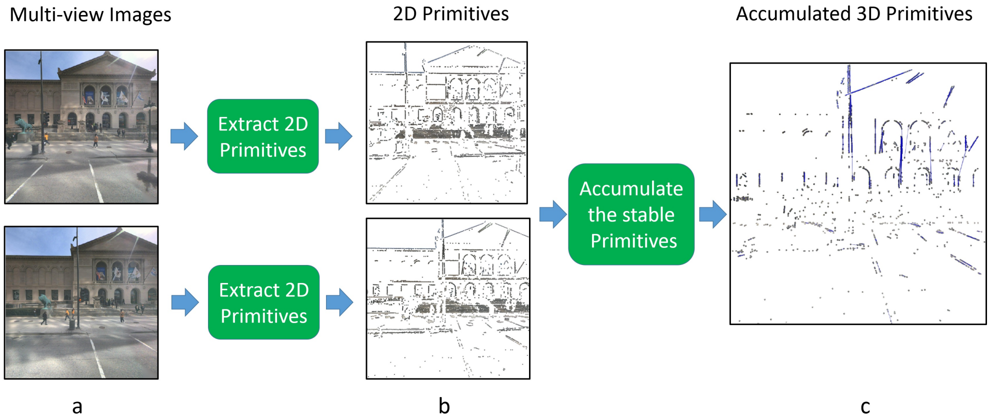


Figure 6.2 Construct the 3D primitives from two-view street-view snapshots.

is dense point clouds which require 3D acquisition device or 3D reconstruction. In contrast, our work use sparse 3D visual primitives that are accumulated from 2D primitives of images. Our RANSAC is described in Algorithm 1.

Algorithm 1: Random sample consensus matching.

```
Compute the match matrix between each observed primitive $\vec{\Pi}_{i=1...N}$ and each model
primitive $\vec{\Pi}_{i=1...M}$: $D_{N\times M}$.
Sort and select the K best matches for each observation primitive → $\hat{D}_{N\times K}$.
for $R$ iterations do
    Randomly select 3 observation primitives from $1\ldots N$ and their correspondences in
    $1\ldots K$ in $\hat{D}_{N\times K}$.
    Estimate the linear 3D transformation (isometry/similarity) T using the Umeyama
    method [141].
    Transform the all $N$ observation primitives to the model space with T.
    Select the geometrically closest matches (within the $K$ best) and compute the match
    score $s$.
    Update the best match ($s_{best}$, $T_{best}$) if necessary.
end for
Return $s_{best}$ and $T_{best}$.
```

Table 6.1 Recognition accuracy for experiments with the KIT object models using median matching (pure chance 0.08%).

Method	*El-Az* 5°	*El-Az* 10°	*El-Az* 20°	*El-Az* 30°	*El-Az* 40°
Med match - Sett. 1	98%	93%	78%	55%	33%
Med match - Sett. 1 (2D)	98%	94%	78%	51%	28%
Med match - Sett. 1 (2D, no acc.)	79%	72%	52%	34%	23%
Med match - Sett. 2	99%	97%	87%	63%	38%
Med match - Shape descr. [34]	88%	75%	47%	33%	19%

Parameter	*Setting 1*	*Setting 2*
Image size	300x300	400x400
Min. energy	0.4	0.4
Max. variance	0.2	0.2
Ext. conf.	0.1	0.1

6.3 Experiments

Our proposed 3D visual primitives-based recognition method was evaluated with both an indoor object dataset and outdoor street-view scenes. The indoor object dataset was the KIT object models web database [60] (see Section 2.4.4), which contains high-quality 3D models of indoor objects. A synthetic view generator was implemented for KIT Objects to generate corresponding 3D models with controlled camera pose. To further evaluate the robustness of our method, we used an outdoor urban scene dataset consisting of 160 street-view images pairs with known camera pose from 4 different cities (see Section 2.4.3).

1. Experiments with KIT object models
   Our experimental results with KIT object models are listed in Table. 6.1. The results showed that using the accumulated 3D visual primitives improved the object recognition accuracy compared with using only 2D visual primitives, and the improvement was clearly visible in the case of relative large viewing angle difference between testing images and their corresponding reference in database. When the viewing angles of test images differed a lot with the corresponding reference in database, the recognition accuracy of both the 3D and

Table 6.2 Recognition accuracy for outdoor urban scenes using median matching.

Three Sets	Set1	Set2	Set3
Number of classes	12	24	40
Number of street-view pairs	48	96	160
By pure chance to find the correct class	8%	4%	2%
Accuracy	92%	80%	75%
The correct class within the best 5 candidates	97%	94%	85%

2D primitive-based approach dropped and it was due to lack of overlaps between the test and reference views. Besides, higher resolution images gave more 3D primitives and more 3D primitives could potentially increase the object recognition accuracy at a cost of heavier computation.

2. Experiments with street-view scenes
Our experimental results with outdoor scene recognition are listed in Table. 6.2. Without any parameter tuning, we applied our recognition algorithm with the realistic outdoor urban scene data. There were 40 urban scenes in our experiments, and each urban scene was considered as a class. For each class, there were 4 pairs of images with moderate occlusion and viewpoint changes. Among the 4 pairs of the images, 1 pair was used for training and the rest 3 pairs were used for testing. Therefore, there were 40 pairs of images in training and 120 pairs in testing. Our experimental results showed that the accumulated 3D primitives together with our RANSAC algorithm produced satisfactory urban scene recognition accuracy for the data with moderate occlusion and viewpoint changes.

6.4 Summary

We propose a 3D visual primitives based recognition method, and it utilizes both 2D appearance and 3D structure from multi-view images. The low level 2D visual primitives are categorized by computational intrinsic dimension, and then they are matched across multi-view images and triangulated to 3D primitives. A simple but effective RANSAC variant is introduced to matching the 3D primitives. Experimental results show that process of accumulation from 2D primitives to 3D primitives improves the object recognition accuracy by selecting more robust primitives. The 3D primitives based approach is more robust for viewpoint changes compared with 2D primitives base approach. Our method achieved good accuracy for the view angle variation up to ±20° with indoor objects dataset and satisfactory accuracy for the urban street-view scenes.

7 CONCLUSIONS

7.1 Summary of the thesis

In this thesis, we target on two research problems: (1) How to estimate the camera pose of a query image with a 3D map consisting of street-view snapshots and point clouds; (2) With an estimated camera pose, how to create meaningful and intuitive applications with a 3D map.

In Chapter 1, we gave the basic theoretical background of the camera pose estimation, demonstrated the wide industry impacts of the camera pose estimation, and explained the objective of this thesis. In Chapter 2, we described the fundamental concepts related to the camera pose estimation and showed the main datasets used in the thesis. In Chapter 3, we presented an in-depth literature review of the camera pose estimation methods.

In Chapter 4, we focused on single-query camera pose estimation with street-view snapshots and point clouds. We systematically investigated *direct* and *indirect* approaches for 6-DoF camera pose estimation, and proposed a *hybrid* approach. We selected and implemented strong baselines for each approach, including one *indirect* feature-based method, one *hybrid* method, two *direct* methods (photometric-based and mutual-information-based methods). Our experiments showed that feature-based approach gave more accurate results than the two *direct* methods when there are at least 4 consistent feature points between query and reference images. Two *direct* methods are sensitive to initialization, however, mutual-information-based method was more robust to environmental changes compared to feature-based and phtometric-based approaches. The *hybrid* method was on par or superior to other evaluated method especially with the challenging Oxford RobotCar dataset.

In Chapter 5, we presented two innovative augmented reality applications by utilizing the camera pose and a 3D map. Firstly, we invented an innovative photo sharing application, where images' geo-meta data are extracted with an indirect pose estimation method and photo sharing experience is improved by utilizing the estimated camera pose and 3D map data. This application shows that the ordinary image sharing experience can be greatly enriched by leveraging global camera pose and 3D map data, because it allows the users to seamlessly transit their viewing points from 2D image space to 3D map space, explore the areas which are not captured in the original image, and navigate in the 3D map. Secondly, we designed an interactive video playback application, where we estimated video frame's camera pose with an indirect method and enriched the video playback with an augmented map. This application adopted a client-server architecture, and users can capture a video with their mobile camera and upload it to the server for post-processing. Global camera pose of the video frames will be returned to the client. During the video playback in the client side, we allow users to expend the field of view to surrounding environments, see the camera motion along the captured scene, and visualize the captured scene in a map view. In Chapter 6, we designed and quantitatively evaluated a 3D method for indoor object and outdoor street scene recognition with the 3D visual primitive. The 3D primitive exploited 3D structure as well as the 2D appearance from the two-view images. This recognition method produced camera pose as an intermediate result.

The research outcomes of this PhD dissertations include 4 publications [36, 37, 38, 40], 5 granted U.S. patents [26, 27, 28, 39, 86] and one open-source project [35]. The scientific novelties of this research work contribute to the industry.

7.2 Future perspectives

Camera pose estimation is one fundamental yet challenging building block for many interesting applications, e.g. autonomous robots, augmented reality, virtual reality etc. If we want to make localization application for a self-driving car, the requirement for the localization accuracy should be in centimeter-level and there can be a few future directions to continue this research.

Firstly, the discussed camera pose estimation problem is mainly to compute the camera pose for a single image with map data, but in the application of autonomous vehicles, we could get video sequences from the cameras mounted on the car. The video sequence contains rich temporal information, and it is definitely worth exploring how the temporal information can improve the camera pose estimation. The visual odometry methods [32, 56, 66, 129] already address this problem to some degree, but even the state-of-the-art visual odometry algorithms drift in long term. Long term visual-based localization remains to be an active and challenging research topic.

Secondly, feature matching can be improved by detecting the dynamic objects and using semantic information. In the feature-based camera pose estimation, if a lot of matched features between images are from dynamic objects, it can cause error in camera pose estimation in spite of using RANSAC [137]. Therefore, if the dynamic objects are detected and excluded from feature matching, the robustness of the feature-based pose estimation method will be improved. Furthermore, feature matching will be more efficient and accurate by considering semantic information [101], because we can give a rule that only the feature points from the same semantic class can be matched between images. The semantic classes can be buildings, traffic signs, roads, trees, etc. For example, features belong to a building in the query image will only be able to match with the features belong to a building in the reference image.

Thirdly, a more compact and invariant reference map can be created to improve the performance of the localization algorithms. This Thesis has explored different localization methods by using street-view snapshots and point clouds, but both the street-view snapshots and point clouds contain many objects in the scene. We know not all of them contribute to localization task. If we know what are the most critical elements for localization, we can build a map that is compact and invariant to certain environmental changes from the street-view snapshots and point clouds. In autonomous vehicle industry, many companies are using either SD-map or HD-map for localization components.

Fourthly, one effective way to improve the localization accuracy is using multiple sensors, such as inertial sensors, radar, camera, GPS, and even LIDAR. In the presence of multiple sensors, Bayesian filters such as Kalman filter and its variants such as extended Kalman filter [114], cubature Kalman filter [4], unscented Kalman filter [143] are good choices to perform the localization tasks.

Last but not least, deep neural network made quite a few breakthroughs in the computer vision and some works target on solving the camera pose estimation [11, 12, 62]. They provide some good performance in small scale camera pose estimation, but these approaches have limitations in large scale. It is worth exploring deep learning based localization methods that work in large scale.

REFERENCES

[1] A. Alahi, R. Ortiz and P. Vandergheynst. FREAK: Fast Retina Keypoint. *IEEE Conference on Computer Vision and Pattern Recognition*. 2012, 510–517.

[2] M. Ambai and Y. Yoshida. CARD: Compact and real-time descriptors. *IEEE International Conference on Computer Vision*. 2011, 97–104.

[3] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, 5297–5307.

[4] I. Arasaratnam and S. Haykin. Cubature kalman filters. *IEEE Transactions on Automatic Control* 54.6 (2009), 1254–1269.

[5] M. A. As' ari, U. U. Sheikh and E. Supriyanto. 3D shape descriptor for object recognition based on Kinect-like depth image. *Image and Vision Computing* 32.4 (2014), 260–269.

[6] H. Bay, T. Tuytelaars and L. Van Gool. Surf: Speeded up robust features. *European Conference on Computer Vision* (2006), 404–417.

[7] S. Benford and G. Giannachi. *Performing mixed reality*. The MIT Press, 2011.

[8] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM* 18.9 (1975), 509–517.

[9] M. Billinghurst. Augmented reality in education. *New horizons for learning* 12.5 (2002), 1–5.

[10] N. C. Blog. *HERE: the next generation of location services*. `https://web.archive.org/web/20130725162540/http://conversations.nokia.com/2012/11/13/here-the-next-generation-of-location-services/`. Accessed: 2021-01-02.

[11] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold and C. Rother. DSAC-differentiable RANSAC for camera localization. *IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 3. 2017.

[12] E. Brachmann and C. Rother. Learning less is more-6d camera localization via 3d surface regression. *IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 8. 2018.

[13] B. Browatzki, V. Tikhanoff, G. Metta, H. H. Bülthoff and C. Wallraven. Active object recognition on a humanoid robot. *IEEE International Conference on Robotics and Automation*. 2012, 2021–2028.

[14] G. C. Burdea and P. Coiffet. *Virtual reality technology*. John Wiley & Sons, 2003.

[15] M. Calonder, V. Lepetit, C. Strecha and P. Fua. Brief: Binary robust independent elementary features. *European Conference on Computer Vision* (2010), 778–792.

[16] F. Camposeco, T. Sattler, A. Cohen, A. Geiger and M. Pollefeys. Toroidal constraints for two-point localization under high outlier ratios. *IEEE Conference on Computer Vision and Pattern Recognition*. 2017, 4545–4553.

[17] O. Chum and J. Matas. Unsupervised discovery of co-occurrence in sparse high dimensional data. *IEEE Conference on Computer Vision and Pattern Recognition*. 2010, 3416–3423.

[18] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 1. 2005, 886–893.

[19] P. Debevec, Y. Yu and G. Borshukov. Efficient view-dependent image-based rendering with projective texture-mapping. *Eurographics Workshop on Rendering Techniques*. Springer. 1998, 105–116.

[20] D. DeTone, T. Malisiewicz and A. Rabinovich. Superpoint: Self-supervised interest point detection and description. *IEEE Conference on Computer Vision and Pattern Recognition Workshop*. 2018, 224–236.

[21] M. Donoser and D. Schmalstieg. Discriminative feature-to-point matching in image-based localization. *IEEE Conference on Computer Vision and Pattern Recognition*. 2014, 516–523.

[22] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii and T. Sattler. D2-net: A trainable cnn for joint description and detection of local features. *IEEE Conference on Computer Vision and Pattern Recognition*. 2019, 8092–8101.

[23] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii and T. Sattler. D2-net: A trainable cnn for joint description and detection of local features. *IEEE Conference on Computer Vision and Pattern Recognition*. 2019, 8092–8101.

[24] J. Engel, V. Koltun and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.3 (2018), 611–625.

[25] J. Engel, T. Schöps and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. *European Conference on Computer Vision*. 2014, 834–849.

[26] L. Fan, J. Fu, K. Roimela and Y. You. *Method and apparatus for determining camera location information and/or camera pose information according to a global coordinate system*. U.S. Patent 9,699,375. 2017.

[27] L. Fan, J. Fu, K. Roimela and Y. You. *Method and apparatus for determining camera location information and/or camera pose information according to a global coordinate system*. U.S. Patent 9,558,559. 2017.

[28] L. Fan, V.-v. Mattila, Y. You, K. Roimela, J. Fu and A. Eronen. *Method and apparatus for determining camera location information and/or camera pose information according to a global coordinate system*. U.S. Patent 10,102,675. 2018.

[29] M. Felsberg and G. Sommer. Image features based on a new approach to 2D rotation invariant quadrature filters. *European Conference on Computer Vision*. 2002, 369–383.

[30] Y. Feng, Z. Zhang, X. Zhao, R. Ji and Y. Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. *IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 264–272.

[31] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24.6 (1981), 381–395.

[32] C. Forster, M. Pizzoli and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. *IEEE International Conference on Robotics and Automation*. 2014, 15–22.

[33] J. H. Friedman, J. L. Bentley and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software* 3.3 (1977), 209–226.

[34] A. Frome, D. Huber, R. Kolluri, T. Bülow and J. Malik. Recognizing objects in range data using regional point descriptors. *European Conference on Computer Vision*. 2004, 224–237.

[35] J. Fu. *Camera Pose Estimation*. `https://github.com/JunshengFu/camera-pose-estimation`. Accessed: 2021-01-02.

[36] J. Fu, L. Fan, K. Roimela, Y. You and V.-V. Mattila. A 3D map augmented photo gallery application on mobile device. *IEEE International Conference on Image Processing*. 2014, 2507–2511.

[37] J. Fu, L. Fan, Y. You and K. Roimela. Augmented and interactive video playback based on global camera pose. *ACM International Conference on Multimedia*. 2013, 461–462.

[38] J. Fu, J.-K. Kämäräinen, A. G. Buch and N. Krüger. Indoor objects and outdoor urban scenes recognition by 3d visual primitives. *Asian Conference on Computer Vision*. Springer. 2014, 270–285.

[39] J. Fu and S. S. Mate. *Method and apparatus for generating a media capture request using camera pose information*. U.S. Patent 9,596,404. 2017.

[40] J. Fu, S. Pertuz, J. Matas and J.-K. Kämäräinen. Performance analysis of single-query 6-DoF camera pose estimation in self-driving setups. *Computer Vision and Image Understanding* 186 (2019), 58–73.

[41] X. S. Gao, X. R. Hou, J. Tang and H. F. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25.8 (2003), 930–943.

[42] A. Geiger, P. Lenz and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. *IEEE Conference on Computer Vision and Pattern Recognition*. 2012.

[43] A. Glent Buch, Y. Yang, N. Kruger and H. Gordon Petersen. In search of inliers: 3d correspondence by local and global voting. *IEEE Conference on Computer Vision and Pattern Recognition*. 2014, 2067–2074.

[44] R. Gomez Ojeda, J. Briales and J. Gonzalez Jimenez. Pl-svo: Semi-direct monocular visual odometry by combining points and line segments. *IEEE International Conference on Intelligent Robots and Systems*. 2016, 4211–4216.

[45] Google. *Glass Enterprise Edition 2*. `https://www.google.com/glass/tech-specs//`. Accessed: 2021-01-02.

[46] Google. *Google Street View*. `https://www.google.com/streetview//`. Accessed: 2021-01-02.

[47] X.-F. Han, J. S. Jin, M.-J. Wang, W. Jiang, L. Gao and L. Xiao. A review of algorithms for filtering the 3D point cloud. *Signal Processing: Image Communication* 57 (2017), 103–112.

[48] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[49] K. He, X. Zhang, S. Ren and J. Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, 770–778.

[50] J. Hecht. Lidar for self-driving cars. *Optics and Photonics News* 29.1 (2018), 26–33.

[51] G. Huang, Z. Liu, G. Pleiss, L. Van Der Maaten and K. Weinberger. Convolutional Networks with Dense Connectivity. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019), 1–1.

[52] P. J. Huber. *Robust statistics*. Springer, 2011.

[53] A. Irschara, C. Zach, J.-M. Frahm and H. Bischof. From structure-from-motion point clouds to fast location recognition. *IEEE Conference on Computer Vision and Pattern Recognition*. 2009, 2599–2606.

[54] A. Iscen, G. Tolias, Y. Avrithis, T. Furon and O. Chum. Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. *IEEE Conference on Computer Vision and Pattern Recognition*. 2017, 2077–2086.

[55] Y. Jiang, J. Yuan and G. Yu. Randomized spatial partition for scene recognition. *European Conference on Computer Vision*. 2012, 730–743.

[56] E. S. Jones and S. Soatto. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *The International Journal of Robotics Research* 30.4 (2011), 407–430.

[57] W. Kabsch. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* 34.5 (1978), 827–828.

[58] T. Kadir and M. Brady. Saliency, scale and image description. *International Journal of Computer Vision* 45.2 (2001), 83–105.

[59] S. Kalkan, F. Wörgötter and N. Krüger. Statistical Analysis of Local 3D Structure in 2D Images. *IEEE Conference on Computer Vision and Pattern Recognition*. 2006, 1114–1121.

[60] A. Kasper, Z. Xue and R. Dillmann. The KIT object models database: An object model database for object recognition, localization and manipulation in service robotics. *The International Journal of Robotics Research* 31.8 (2012), 927–934.

[61] S. Katz, A. Tal and R. Basri. Direct visibility of point sets. *ACM SIGGRAPH*. 2007, 24–es.

[62] A. Kendall, M. Grimes and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. *IEEE international Conference on Computer Vision*. 2015, 2938–2946.

[63] A. Khurshid, S. Cleger and R. Grunitzki. A scene classification approach for augmented reality devices. *International Conference on Human-Computer Interaction*. Springer. 2020, 164–177.

[64] H. Kim, D. Lee, T. Oh, S. W. Lee, Y. Choe and H. Myung. Feature-based 6-DoF camera localization using prior point cloud and images. *Robot Intelligence Technology and Applications 2*. Springer, 2014, 3–11.

[65] H. Kim, D. Lee, T. Oh, S. W. Lee, Y. Choe and H. Myung. Feature-based 6-DoF camera localization using prior point cloud and images. *Robot Intelligence Technology and Applications 2*. Springer, 2014, 3–11.

[66] B. Kitt, A. Geiger and H. Lategahn. Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme. *IEEE Intelligent Vehicles Symposium*. 2010, 486–492.

[67] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. *IEEE and ACM International Symposium on Mixed and Augmented Reality*. 2007, 225–234.

[68] N. Kruger, P. Janssen, S. Kalkan, M. Lappe, A. Leonardis, J. Piater, A. J. Rodriguez-Sanchez and L. Wiskott. Deep Hierarchies in the Primate Visual Cortex: What Can We Learn for Computer Vision?: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8 (2013), 1847–1871.

[69] S. Lazebnik, C. Schmid and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 2. 2006, 2169–2178.

[70] S. Leutenegger, M. Chli and R. Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. *IEEE International Conference on Computer Vision*. 2011, 2548–2555.

[71] L.-J. Li, H. Su, Y. Lim and L. Fei-Fei. Object bank: An object-level image representation for high-level visual recognition. *International Journal of Computer Vision* 107.1 (2014), 20–39.

[72] Y. Li, N. Snavely and D. P. Huttenlocher. Location recognition using prioritized feature matching. *European conference on computer vision*. 2010, 791–804.

[73] Y. Liu, Q. Chen, W. Chen and I. Wassell. Dictionary learning inspired deep network for scene recognition. *AAAI Conference on Artificial Intelligence*. Vol. 32. 1. 2018.

[74] D. G. Lowe. Object recognition from local scale-invariant features. *IEEE International Conference on Computer Vision*. Vol. 2. 1999, 1150–1157.

[75] R. Ltd. *RouteShoot*. `https://apps.apple.com/us/app/routeshoot/id567458803`. Accessed: 2021-01-03.

[76] G. Lu, Y. Yan, L. Ren, J. Song, N. Sebe and C. Kambhamettu. Localize me anywhere, anytime: a multi-task point-retrieval approach. *IEEE International Conference on Computer Vision*. 2015, 2434–2442.

[77] B. D. Lucas, T. Kanade et al. An iterative image registration technique with an application to stereo vision. Vancouver. 1981.

[78] H. Luo, Y. Yang, B. Tong, F. Wu and B. Fan. Traffic sign recognition using a multi-task convolutional neural network. *IEEE Transactions on Intelligent Transportation Systems* 19.4 (2017), 1100–1111.

[79] J. Luo, D. Joshi, J. Yu and A. Gallagher. Geotagging in multimedia and computer vision—a survey. *Multimedia Tools and Applications* 51.1 (2011), 187–211.

[80] Z. Lv, A. Halawani, S. Feng, S. Ur Réhman and H. Li. Touch-less interactive augmented reality game on vision-based wearable device. *Personal and Ubiquitous Computing* 19.3 (2015), 551–567.

[81] W. Maddern, M. Milford and G. Wyeth. CAT-SLAM: probabilistic localisation and mapping using a continuous appearance-based trajectory. *The International Journal of Robotics Research* 31.4 (2012), 429–451.

[82] W. Maddern, G. Pascoe, C. Linegar and P. Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research* 36.1 (2017), 3–15.

[83] W. Maddern, A. Stewart, C. McManus, B. Upcroft, W. Churchill and P. Newman. Illumination invariant imaging: Applications in robust vision-based localisation, mapping and classification for autonomous vehicles. *IEEE International Conference on Robotics and Automation Workshop*. Vol. 2. 2014, 3.

[84] V. Mani and D. Rivazhagan. Survey of Medical Image Registration. *Journal of Biomedical Engineering and Technology* 1.2 (2013), 8–25.

[85] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy and D. Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. *arXiv*. 2020.

[86] S. S. Mate, J. Leppanen, J. Fu and P. Babahajiani. *Device with an adaptive camera array*. U.S. Patent 9,996,943. 2018.

[87] A. F. McDaid, D. Greene and N. Hurley. Normalized mutual information to evaluate overlapping community finding algorithms. *arXiv preprint arXiv:1110.2515* (2011).

[88] C. McManus, W. Churchill, W. Maddern, A. D. Stewart and P. Newman. Shady dealings: Robust, long-term visual localisation using illumination invariance. *IEEE International Conference on Robotics and Automation*. 2014, 901–906.

[89] Microsoft. *Microsoft HoloLens*. `https://docs.microsoft.com/en-us/hololens/`. Accessed: 2021-01-02.

[90] Microsoft. *Microsoft Pix*. `https://www.microsoft.com/en-us/research/blog/new-microsoft-pix-features-let-take-bigger-wider-pictures-turns-videos-comics/`. Accessed: 2021-01-02.

[91] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision* 60.1 (2004), 63–86.

[92] M. J. Milford and G. F. Wyeth. SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights. *IEEE International Conference on Robotics and Automation*. 2012, 1643–1649.

[93] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser and P. Sayd. Generic and real-time structure from motion using local bundle adjustment. *Image and Vision Computing* 27.8 (2009), 1178–1193.

[94] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. *International Conference on Computer Vision Theory and Applications* 2.331-340 (2009), 2.

[95] R. Mur-Artal, J. M. M. Montiel and J. D. Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics* 31.5 (2015), 1147–1163.

[96] R. Mur-Artal and J. D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics* 33.5 (2017), 1255–1262.

[97] T. Naseer, L. Spinello, W. Burgard and C. Stachniss. Robust visual robot localization across seasons using network flows. *AAAI Conference on Artificial Intelligence*. Vol. 28. 1. 2014.

[98] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. *IEEE International Aymposium on Mixed and Augmented Reality*. 2011, 127–136.

[99] R. A. Newcombe, S. J. Lovegrove and A. J. Davison. DTAM: Dense tracking and mapping in real-time. *IEEE International Conference on Computer Vision*. 2011, 2320–2327.

[100] D. Nistér, O. Naroditsky and J. Bergen. Visual odometry. *IEEE Conference on Computer Vision and Pattern Recognition*. 2004.

[101] H. Noh, S. Hong and B. Han. Learning deconvolution network for semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognitio*. 2015, 1520–1528.

[102] Y. Ohta and H. Tamura. *Mixed reality: merging real and virtual worlds*. Springer Publishing Company, Incorporated, 2014.

[103] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision* 42.3 (2001), 145–175.

[104] Y. Ono, E. Trulls, P. Fua and K. M. Yi. LF-Net: Learning local features from images. *arXiv preprint arXiv:1805.09662* (2018).

[105] C. Papazov and D. Burschka. An efficient ransac for 3d object recognition in noisy and occluded scenes. *Asian Conference on Computer Vision*. 2010, 135–148.

[106] S. Park, J. Kim, R. Mizouni and U. Lee. Motives and concerns of dashcam video sharing. *CHI Conference on Human Factors in Computing Systems*. 2016, 4758–4769.

[107] G. Pascoe, W. Maddern, M. Tanner, P. Pinies and P. Newman. NID-SLAM: Robust Monocular SLAM using Normalised Information Distance. *IEEE Conference on Computer Vision and Pattern Recognition*. July 2017.

[108] J. Philbin, O. Chum, M. Isard, J. Sivic and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. *IEEE Conference on Computer Vision and Pattern Recognition*. 2007, 1–8.

[109] A. P. Pozo, M. Toksvig, T. F. Schrager, J. Hsu, U. Mathur, A. Sorkine-Hornung, R. Szeliski and B. Cabral. An integrated 6DoF video camera and system design. *ACM Transactions on Graphics* 38.6 (2019), 1–16.

[110] N. Pugeault, F. Wörgötter and N. Krüger. Accumulated Visual Representation for Cognitive Vision. *British Machine Vision Conference*. 2008, 1–10.

[111] C. R. Qi, H. Su, K. Mo and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*. 2017, 652–660.

[112] F. Radenović, G. Tolias and O. Chum. CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples. *European Conference on Computer Vision*. 2016.

[113] S. Ray. *Applied photographic optics*. Routledge, 2002.

[114] K. Reif, S. Gunther, E. Yaz and R. Unbehauen. Stochastic stability of the discrete-time extended Kalman filter. *IEEE Transactions on Automatic Control* 44.4 (1999), 714–728.

[115] E. Rodolà, A. Albarelli, F. Bergamasco and A. Torsello. A scale independent selection process for 3d object recognition in cluttered scenes. *International journal of computer vision* 102.1-3 (2013), 129–145.

[116] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. *European Conference on Computer Vision* (2006), 430–443.

[117] E. Rublee, V. Rabaud, K. Konolige and G. Bradski. ORB: An efficient alternative to SIFT or SURF. *IEEE International Conference on Computer Vision*. 2011, 2564–2571.

[118] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747* (2016).

[119] R. B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). *IEEE International Conference on Robotics and Automation*. 2011, 1–4.

[120] T. Sattler, M. Havlena, K. Schindler and M. Pollefeys. Large-Scale Location Recognition and the Geometric Burstiness Problem. *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, 1582–1590.

[121] T. Sattler, B. Leibe and L. Kobbelt. Fast image-based localization using direct 2d-to-3d matching. *IEEE International Conference on Computer Vision*. 2011, 667–674.

[122] T. Sattler, B. Leibe and L. Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.9 (2016), 1744–1756.

[123] N. Savinov, A. Seki, L. Ladicky, T. Sattler and M. Pollefeys. Quad-networks: unsupervised learning to rank for interest point detection. *IEEE Conference on Computer Vision and Pattern Recognition*. 2017, 1822–1830.

[124] J. L. Schonberger and J.-M. Frahm. Structure-from-motion revisited. *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, 4104–4113.

[125] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. *IEEE International Conference on Computer Vision*. 2015, 118–126.

[126] K. Simonyan, A. Vedaldi and A. Zisserman. Learning local feature descriptors using convex optimisation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.8 (2014), 1573–1585.

[127] N. Snavely, S. M. Seitz and R. Szeliski. Photo tourism: exploring photo collections in 3D. *ACM SIGGRAPH*. 2006, 835–846.

[128] N. Snavely, S. M. Seitz and R. Szeliski. Modeling the world from internet photo collections. *International journal of computer vision* 80.2 (2008), 189–210.

[129] A. Solin, S. Cortes, E. Rahtu and J. Kannala. Inertial odometry on handheld smartphones. *IEEE International Conference on Information Fusion*. 2018, 1–5.

[130] Y. Song, X. Chen, X. Wang, Y. Zhang and J. Li. 6-DOF image localization from massive geo-tagged reference images. *IEEE Transactions on Multimedia* 18.8 (2016), 1542–1554.

[131] L. Svärm, O. Enqvist, F. Kahl and M. Oskarsson. City-Scale Localization for Cameras with Known Vertical Direction. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.7 (2017), 1455–1461.

[132] B. Templeton. *Elon Musk's War On LIDAR: Who Is Right And Why Do They Think That?* `https://www.forbes.com/sites/bradtempleton/2019/05/06/elon-musks-war-on-lidar-who-is-right-and-why-do-they-think-that/`. Accessed: 2021-01-02.

[133] C. Toft, E. Stenborg, L. Hammarstrand, L. Brynte, M. Pollefeys, T. Sattler and F. Kahl. Semantic match consistency for long-term visual localization. *European Conference on Computer Vision*. 2018, 383–399.

[134] E. Tola, V. Lepetit and P. Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.5 (2010), 815–830.

[135] A. Torii, H. Taira, J. Sivic, M. Pollefeys, M. Okutomi, T. Pajdla and T. Sattler. Are Large-Scale 3D Models Really Necessary for Accurate Visual Localization?: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.3 (2021), 814–829.

[136] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi and T. Pajdla. 24/7 place recognition by view synthesis. *IEEE Conference on Computer Vision and Pattern Recognition*. 2015, 1808–1817.

[137] P. H. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding* 78.1 (2000), 138–156.

[138] T. Tuytelaars and K. Mikolajczyk. *Local invariant feature detectors: a survey*. Now Publishers Inc, 2008.

[139] T. Tykkälä, A. I. Comport and J.-K. Kämäräinen. Photorealistic 3D mapping of indoors by RGB-D scanning process. *IEEE International Conference on Intelligent Robots and Systems*. 2013, 1050–1055.

[140] S. Ullman. The interpretation of structure from motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences* 203.1153 (1979), 405–426.

[141] S. Umeyama. Least-Squares Estimation of Transformation Parameters Between Two Point Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13 (1991), 376–380.

[142] P. Viola and W. M. Wells III. Alignment by maximization of mutual information. *International Journal of Computer Vision* 24.2 (1997), 137–154.

[143] E. A. Wan, R. Van Der Merwe and S. Haykin. The unscented Kalman filter. *Kalman Filtering and Neural Networks* 5.2007 (2001), 221–280.

[144] X. Wang, W. Dong, M. Zhou, R. Li and H. Zha. Edge Enhanced Direct Visual Odometry. *British Machine Vision Conference*. 2016.

[145] C. Wu. Towards Linear-Time Incremental Structure from Motion. *International Conference on 3D Vision*. 2013, 127–134.

[146] C. Wu, S. Agarwal, B. Curless and S. M. Seitz. Multicore bundle adjustment. *IEEE Conference on Computer Vision and Pattern Recognition*. 2011, 3057–3064.

[147] R. Wu, B. Wang, W. Wang and Y. Yu. Harvesting discriminative meta objects with deep CNN features for scene classification. *IEEE International Conference on Computer Vision*. 2015, 1287–1295.

[148] Y. Xiang, W. Choi, Y. Lin and S. Savarese. Data-driven 3d voxel patterns for object category recognition. *IEEE Conference on Computer Vision and Pattern Recognition*. 2015, 1903–1911.

[149] L. Xie, F. Lee, L. Liu, K. Kotani and Q. Chen. Scene recognition: A comprehensive survey. *Pattern Recognition* 102 (2020), 107205.

[150] L. Zhang and S. Rusinkiewicz. Learning to detect features in texture images. *IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 6325–6333.

[151] Z. Zhang. Microsoft kinect sensor and its effect. *IEEE Multimedia* 19.2 (2012), 4–10.

[152] H. Zhao, M. Tang and H. Ding. HoPPF: A novel local surface descriptor for 3D object recognition. *Pattern Recognition* 103 (2020), 107272.

[153] W. Zhao, D. Nister and S. Hsu. Alignment of continuous video onto 3d point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.8 (2005), 1305–1318.

[154] K. Zhou, Q. Hou, R. Wang and B. Guo. Real-time kd-tree construction on graphics hardware. *ACM Transactions on Graphics* 27.5 (2008), 1–11.

PUBLICATIONS

PUBLICATION

I

Performance analysis of single-query 6-DoF camera pose estimation in self-driving setups

J. Fu, S. Pertuz, J. Matas and J.-K. Kämäräinen

Computer Vision and Image Understanding 186.(2019), 58–73

DOI: `https://doi.org/10.1016/j.cviu.2019.04.009`

Computer Vision and Image Understanding 186 (2019) 58–73

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Performance analysis of single-query 6-DoF camera pose estimation in self-driving setups☆

Junsheng Fu [a],[*], Said Pertuz [a,b], Jiri Matas [c], Joni-Kristian Kämäräinen [a]

[a] *Tampere University - Hervannan Campus, Department of Signal Processing, P.O. Box 553, FI-33101 Tampere, Finland*
[b] *Universidad Industrial de Santander, 680003 Bucaramanga, Colombia*
[c] *Czech Technical University in Prague, Faculty of Electrical Engineering, Technicka 2, 16627 Praha 6, Czech Republic*

ARTICLE INFO

Communicated by: P Mordohai

Keywords:
Camera pose estimation
3D point cloud
Hybrid method
Photometric matching
Mutual information
Self driving car

ABSTRACT

In this work, we consider the problem of single-query 6-DoF camera pose estimation, i.e. estimating the position and orientation of a camera by using reference images and a point cloud. We perform a systematic comparison of three state-of-the-art strategies for 6-DoF camera pose estimation: feature-based, photometric-based and mutual-information-based approaches. Two standard datasets with self-driving setups are used for experiments, and the performance of the studied methods is evaluated in terms of success rate, translation error and maximum orientation error. Building on the analysis of the results, we evaluate a hybrid approach that combines feature-based and mutual-information-based pose estimation methods to benefit from their complementary properties for pose estimation. Experiments show that (1) in cases with large appearance change between query and reference, the hybrid approach outperforms feature-based and mutual-information-based approaches by an average increment of 9.4% and 8.7% in the success rate, respectively; (2) in cases where query and reference images are captured at similar imaging conditions, the hybrid approach performs similarly as the feature-based approach, but outperforms both photometric-based and mutual-information-based approaches with a clear margin; (3) the feature-based approach is consistently more accurate than mutual-information-based and photometric-based approaches when at least 4 consistent matching points are found between the query and reference images.

1. Introduction

Camera pose estimation is a fundamental technology for various applications, such as augmented reality (Taylor, 2016), virtual reality (Ohta and Tamura, 2014), and robotic localization (Castellanos and Tardos, 2012). The aim of 6 degrees of freedom (DoF) camera pose estimation is to find the 3-DoF location and 3-DoF orientation of the query image in a given reference coordinate system. In the literature, the classical approach for 6-DoF camera pose estimation is to register a 2D query image with previously acquired reference data, which often consist of a set of reference images and corresponding 3D point clouds. In practice, this is a fundamental yet challenging problem due to large displacements between the query and reference images, as well as image variations caused by changes in the appearance of the scenes, weather and lighting conditions (Maddern et al., 2017; Mishkin et al., 2015). Depending on how the 6-DoF pose estimation problem is solved, state-of-the-art methods can be divided into 2 main categories: *direct* and *indirect* approaches. In our scope, *direct* approach means that the 6-DoF camera pose is directly optimized by a cost function defined over the 6D pose space. For example, the 6-DoF camera pose can be computed by directly minimizing a cost function that compares the query image with a rendered synthetic view from a 3D point cloud (Pascoe et al., 2017; Tykkälä et al., 2013; Newcombe et al., 2011a,b).

In the *indirect* approach, the query image is registered to the 3D point cloud by matching point features extracted in the query image and the reference images (Mishkin et al., 2015; Song et al., 2016; Irschara et al., 2009; Kim et al., 2014), and the reference images and the 3D point cloud are defined in the same world coordinate system. Both *direct* and *indirect* approaches have shown good performance in previous works with different datasets and experimental settings (Pascoe et al., 2017; Mishkin et al., 2015; Song et al., 2016). However, the relative performance of the *direct* and *indirect* approaches have not been studied in the same working conditions with large-scale, realistic datasets.

Although both the *indirect* and *direct* approaches have been widely utilized for 6-DoF pose estimation, we have identified two important

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.cviu.2019.04.009.

[*] Corresponding author.
E-mail address: junsheng.fu@tut.fi (J. Fu).

https://doi.org/10.1016/j.cviu.2019.04.009
Received 22 May 2018; Received in revised form 20 March 2019; Accepted 17 April 2019
Available online 25 April 2019

questions that warrant further research: first, there is no consensus in the community about which strategies yield the best performance in real-life conditions, where the appearance of the reference and query images change significantly according to different weather, lighting and season conditions. Second, in the literature, pose estimation strategies are often assessed as a part of full pipelines that involve additional pre- or post-processing steps, *e.g.* the incorporation of information from previous poses in sequential data or global optimization strategies in simultaneous localization and mapping approaches. As a result, the contribution of pose estimation methods on the overall performance of the system, as well as their response to different imaging factors, remains unclear. In order to tackle the aforementioned problems, we implemented and studied three state-of-the-art camera pose estimation approaches for the estimation of 6-DoF camera pose of a single query image using reference images and a point cloud. Specifically, the three implementations are one *indirect* approach, a feature-based method in Kim et al. (2014), and two *direct* approaches: a photometric-based method (Tykkälä et al., 2013) and a mutual-information-based method (Pascoe et al., 2017). The motivation for studying the selected methods is that they are state-of-the-art, have good speed performance and can be conveniently implemented and tested in the same conditions (Pascoe et al., 2017; Tykkälä et al., 2013; Kim et al., 2014).

We perform a systematic and extensive experimental comparison of the studied approaches and analyze their performances. Based on the obtained results, we evaluate a hybrid approach that combines the feature-based and mutual information-based camera pose estimation methods, and present an architecture for computing the 6-DoF camera pose from rough 2-DoF spatial position estimates. As the **main contribution** of this work, we perform an extensive comparison and analysis of three strategies for 6-DoF camera pose estimation: a feature-based approach, a photometric-based approach, and a mutual-information-based approach. We find that the feature-based approach is more accurate than photometric-based and mutual-information-based approaches with as few as 4 consistent feature points between the query and reference images. However, the mutual-information-based approach is often more robust and can provide a pose estimate when the feature-based approach fails. We experimentally demonstrate that a hybrid approach, which combines the feature- and mutual-information-based approaches, outperforms both. All source code for camera pose estimation methods and their performance evaluation will be made publicly available.[1]

In addition, we study the performance of the hybrid approach with an architecture that allows computing camera pose with multiple reference images and allows to naturally integrate and refine pose priors in large uncertainty cases. For the experiments, we used two publicly available datasets: the KITTI dataset (Geiger et al., 2012) and Oxford RobotCar dataset (Maddern et al., 2017). The KITTI dataset provides 11 individual sequences with ground truth trajectories. The recently released Oxford RobotCar dataset (Maddern et al., 2017) contains many repetitions on the same route. RobotCar dataset provides different combinations of weather, traffic and pedestrians, with long-term changes such as construction and roadworks, which allows a more challenging evaluation in realistic conditions. Our comparison shows how the hybrid approach outperforms feature-based, photometric-based or mutual-information-based approaches. Furthermore, the experiments show that using multiple reference images improves the robustness of all pose estimation pipelines.

1.1. Related work

Camera pose estimation using vision has received significant attention in recent decades. We focus on the case of registering a single query image with one or several reference images and 3D point clouds. The approaches can be divided into 2 main categories: *indirect* approaches (Irschara et al., 2009; Kim et al., 2014; Klein and Murray, 2007; Geiger et al., 2011; Kitt et al., 2010) and *direct* approaches (Pascoe et al., 2017; Tykkälä et al., 2013; Newcombe et al., 2011a). It is important to notice that many of the above works introduce a Simultaneous Localization and Mapping (SLAM) method. Specifically, camera pose estimation discussed in this paper is only one component utilized within more complex SLAM methods. In our discussion we refer only to the camera pose estimation part of them.

The *indirect* approaches establish 2D-3D correspondences between the query image and the 3D point cloud. The reference images and the 3D point cloud are pre-registered, so the 2D-3D correspondences are indirectly obtained by establishing 2D-2D correspondences between the query image and the reference images. Specifically, the query image is registered with the reference images by utilizing feature detectors for finding salient image structures for localization, e.g. corners (Rosten and Drummond, 2006; Mikolajczyk and Schmid, 2004), blobs (Lowe, 1999; Bay et al., 2006; Kadir and Brady, 2001) or regions (Matas et al., 2004; Tuytelaars and Van Gool, 2000, 2004; Mori et al., 2004). Then feature descriptors (Calonder et al., 2010; Rublee et al., 2011; Leutenegger et al., 2011; Alahi et al., 2012; Lowe, 1999; Bay et al., 2006; Dalal and Triggs, 2005; Tola et al., 2010; Ambai and Yoshida, 2011) are used to provide a robust representation regardless of appearance changes due to different viewpoints, weather, lighting, etc. Given the set of 2D-3D correspondences, a Perspective-n-Point solver (Torr and Zisserman, 2000; Gao et al., 2003) and RANSAC (Fischler and Bolles, 1981; Torr and Zisserman, 2000) are applied to compute the relative 6-DoF camera pose between the query image and the reference 3D point cloud. Because different combinations of 2D-3D correspondences lead to different camera pose estimations, the *indirect* approach can be considered as a combinatorial optimization method. A few of the popular *indirect* methods are described as follows: PTAM (Klein and Murray, 2007) is a widely used featured-based monocular SLAM algorithm that allows robust state estimation in real-time. LIBVISO1 (Kitt et al., 2010) is a feature-based 6 DoF camera pose estimation method for a stereo camera, and it is extended into LIBVISO2 (Geiger et al., 2011) which supports monocular ego-motion estimation. Besides, 3D scene representation either from LIDAR or Structure-from-Motion pipelines can be utilized to estimate the camera pose. One work (Irschara et al., 2009) registers on-line images to a sparse 3D scene generated by Structure-from-Motion pipelines. Another work (Kim et al., 2014) estimates camera pose by using LIDAR point cloud and reference images.

The *direct* approaches compute the 6-DoF camera pose by minimizing a cost function directly in the 6D space of camera poses (Pascoe et al., 2017; Tykkälä et al., 2013; Newcombe et al., 2011a,b; Engel et al., 2014, 2018), and do not need to extract local features of images. One commonly used cost function is the photometric error between the query image and the reference view, where the reference view is generated from the reference 3D point cloud (Tykkälä et al., 2013; Newcombe et al., 2011a,b). The *direct* photometric-based methods usually have good speed performance. For example, LSD-SLAM (Engel et al., 2014) is a monocular SLAM which allows to build large-scale maps of the environment and runs in real-time on a CPU. The recent DSO (Engel et al., 2018) combines a fully direct probabilistic model with joint optimization of all model parameters and it can be achieved in real-time by omitting the smoothness prior used in other direct methods and instead sampling pixels evenly throughout the images. However, they are arguably less robust to real-world global illumination changes (Newcombe et al., 2011b). A recent work (Pascoe et al., 2017) utilizes a mutual-information-based cost function for *direct* 6-DoF camera pose estimation outperforming both the feature-based and photometric-based approaches in two challenging datasets with large image variations. This mutual-information-based approach has been tailored for the SLAM problem and it relies on a well-initialized reference image (Pascoe et al., 2017). However, it is still unclear what the performance of the mutual-information-based approach would be

[1] https://github.com/JunshengFu/camera-pose-estimation.

without accounting for the initialization problem, where single query image is to be registered with no prior on the pose.

Besides the *direct* and *indirect* approaches, a semi-direct visual odometry pipeline, the SVO, has been proposed by Forster et al. (2014). In SVO, feature-correspondences are an implicit result of direct motion estimation rather than of explicit feature extraction and matching. Thus, feature extraction is only required when the initial key frame is selected to initialize the construction of a new 3D point cloud. The advantage of this approach is its increased speed due to the lack of feature-extraction at every frame and increased accuracy through sub-pixel feature correspondence. After the feature correspondences and an initial estimate of the camera pose are established, the algorithm continues using only point-features. In this work, we are interested in the solution of the single-query pose estimation problem. The SVO approach is designed for solving the pose estimation problem in the context of multiple, sequential frames and has therefore not been considered in this work.

A recent work (Delmerico and Scaramuzza, 2018) compares visual-inertial odometry algorithms on different hardware configurations, but their focus is on monocular visual-inertial odometry methods. Another benchmark (Li et al., 2016) provides detailed performance analysis of open source visual SLAM pipelines on different datasets. However, their work is focused on comparing the performance of the whole visual SLAM pipeline instead of a single step such as the pose estimation. To the best of our knowledge, there is a lack of prior art comparing the stand alone performance of *direct* and *indirect* camera pose estimation approaches in this scenario.

1.2. Overview

Based on our literature review, we selected and implemented three state-of-the-art 6-DoF pose estimation methods: (1) an *indirect* feature-based method (Kim et al., 2014), (2) a *direct* photometric-based method (Tykkälä et al., 2013) and (3) a *direct* mutual-information-based method (Pascoe et al., 2017). We choose these 3 approaches because they provide good performance and can be adapted for the same experimental settings. The details of these methods are presented in Section 2. In order to conduct a rigorous and systematic analysis of their practical performance, the studied methods were compared in three different scenarios: the single-reference case, the multi-reference case and the large uncertainty case. Each one of the experimental setups for these 3 scenarios are described in Section 3. Experiments and results on real datasets are presented in Section 4. Based on the experimental results, we also evaluate a *hybrid approach* that combines *direct* and *indirect* methods for an improved performance. The conclusions and the implementation details of this work are presented in Section 5 and Appendices, respectively.

2. Evaluated pose estimation methods

The methods selected for comparison in this work are representative examples of *direct* and *indirect* approaches with state-of-the-art performance. In this section, we describe each one of the methods in the simplest scenario, where the inputs are a query image I_Q, and a single *reference tuple* (I_R, P_R) that is formed by a reference image I_R and its registered 3D point cloud P_R (see Fig. 1). The aim is to find the 6D pose of the query image I_Q.

2.1. Indirect feature-based (FB) pose estimation

Standard feature-based pose estimation can be divided into four main steps: (1) feature detection, (2) feature matching, (3) grouping of 2D-3D correspondences, and (4) Perspective-n-Point pose estimation. The block diagram of the feature-based (FB) method is shown in Fig. 2. In the first step, a feature detector and a feature descriptor are applied to both query and reference images to detect points – or regions – of interest and compute descriptors from pixels surrounding each point of interest. Secondly, based on the previously computed descriptors, 2D-2D point correspondences are sought between query and reference images by means of feature matching. Thirdly, since the 3D point cloud is registered with the reference image, the 2D-3D correspondences between the query image and the 3D point cloud can be established indirectly through the 2D-2D correspondences between points of interest in the query and reference images. Finally, a Perspective-n-Point solver (Gao et al., 2003) and RANSAC (Fischler and Bolles, 1981; Torr and Zisserman, 2000) are applied for computing the 6-DoF camera pose from these 2D-3D correspondences. The algorithm and implementation details of each stage of the feature-based pose estimation can be found in Appendix A.

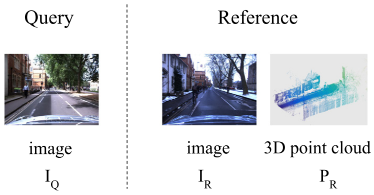


Fig. 1. Inputs for the pose estimation methods in the simplest scenario: a query image I_Q and a *reference tuple* (I_R, P_R), where I_R is a single reference image and P_R is the registered 3D point cloud associated to I_R. Both the point cloud P_R and the camera pose of the reference image I_R are defined in a common world coordinate system. The aim is to estimate the 6D pose of the query image I_Q.

2.2. Direct photometric-based (PB) pose estimation

The *direct* photometric-based approach (Tykkälä et al., 2013) is defined as a direct minimization of a cost function defined over the 6D space of camera poses. The pixel intensities of the query image and a rendered synthetic view from the 3D point cloud are directly compared in the cost function (Tykkälä et al., 2013). The photometric-based approach can be divided into three main steps: (1) synthetic image generation, (2) photometric matching, and (3) coarse-to-fine search.

The block diagram of this method is shown in Fig. 3. In summary the algorithm works as follows: firstly, for rendering purposes, a *colored 3D point cloud* is generated by projecting each 3D point of the cloud P_R to the reference image frame and then assigning the colors from the reference image at that location. Subsequently, we generate a synthetic image I_S by projecting the colored 3D point cloud into an image plane (see Appendix B.1), where the transformation matrix $\mathbf{M}$ of the reference image is used as the initial pose estimate. The goal is to find the transformation matrix that minimizes the photometric error between the synthetic view and the query image:

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} RES(I_Q, I_S), \tag{1}$$

where $RES(\cdot,\cdot)$ is the residual function used to compute the photometric error.

In this work, we solve (1) by means of a coarse-to-fine grid search (see Appendix B.3). It should be noted that in common tracking applications where the transformation baseline is small, fast optimization can be implemented by using Jacobian and gradient-based optimization (Tykkälä et al., 2013). However, in the case of big appearance differences between the query and reference images, gradient-based optimization often fails to find global solutions, so we adopted a grid search in our experiments. A more detailed description of the photometric pose estimation method with implementation details can be found in Appendix B.

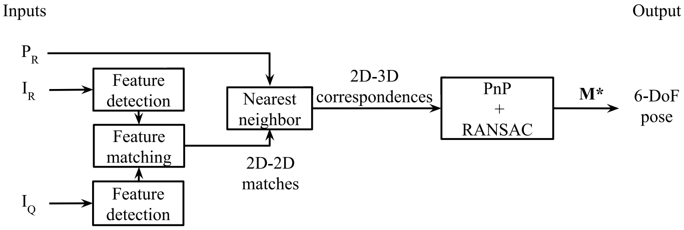


Fig. 2. Block diagram of feature-based camera pose estimation. I_Q is the query image. The reference image I_R and the 3D point cloud P_R are pre-registered and defined in the world coordinate system. $\mathbf{M}^*$ is the estimated transformation matrix. For the detailed descriptions of each step see Appendix A.

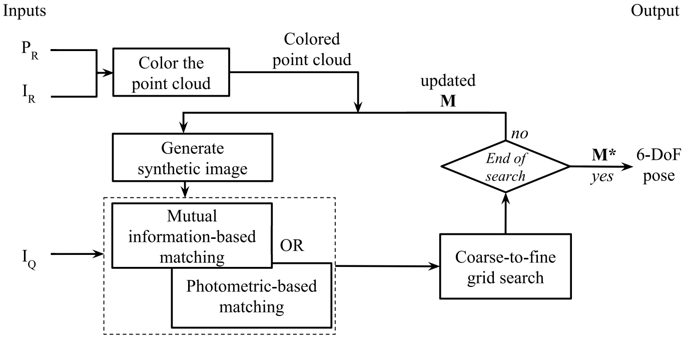


Fig. 3. Block diagram of direct photometric-based and mutual-information-based camera pose estimation. I_Q is the query image. The reference image I_R and the 3D point cloud P_R are pre-registered and defined in the world coordinate system. $\mathbf{M}^*$ is the estimated transformation matrix. For the detailed descriptions of each step see Appendix B.

2.3. Direct mutual-information-based (MI) pose estimation

The *direct* mutual-information-based approach is similar to the photometric-based approach presented in previous section with the main difference being that, in the cost function (1), the *normalized mutual information* (NMI) is used instead of the photometric error. Specifically, the mutual information-based pose estimation problem is formulated as the minimization problem:

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} 1 - NMI(I_Q, I_S), \quad (2)$$

where $\mathbf{M}^*$ is the estimated camera pose, I_Q is the query image, I_S is the synthetic image; and the Normalized Mutual Information (NMI) is computed as (McDaid et al., 2011):

$$NMI(I_S, I_Q) = \frac{MI(I_S, I_Q)}{max(H(I_S), H(I_Q))} \quad (3)$$

with

$$MI(I_S, I_Q) = H(I_S) + H(I_Q) - H(I_S, I_Q) \ , \quad (4)$$

where $H(I_S, I_Q)$ is the joint entropy of I_S and I_Q, $H(I_S)$ and $H(I_Q)$ are the marginal entropies of I_S and I_Q, and $MI(I_S, I_Q)$ is the mutual information between I_S and I_Q.

2.4. Hybrid (HY) pose estimation

In our experiments, we also evaluate a combination of indirect and direct approaches for pose estimation. This approach is inspired by the strong empirical evidence in our experiments showing that: (1) the feature-based method is superior in accuracy if a sufficient number of matches can be found (see details in Sections 4.3 and 4.5); (2) the mutual-information-based approach can still provide a reasonable estimate in cases where the feature-based method fails to generate an estimate (no enough matched features found between the reference and query images). Therefore, our hybrid approach first executes the feature-based method and, if it fails to compute at least 4 consistent matching points between the query and reference images, then it switches to the MI-based method.

Specifically, given one query image I_Q and one *reference tuple* (I_R, P_R), a feature detector is firstly applied to both the query image I_Q and the reference image I_R, and then we apply feature matching to obtain 2D-2D matched features. Since the point cloud P_R is registered with the reference image I_Q, the 2D-3D correspondences can be found indirectly. Then a PnP solver (Gao et al., 2003) and RANSAC (Torr and Zisserman, 2000) are applied to the 2D-3D correspondences. For the PnP solver (Gao et al., 2003), at least 4 consistent 2D-3D correspondence pairs are required. If the camera pose of the query image cannot be estimated due to less than four 2D-3D correspondences (Torr and Zisserman, 2000; Gao et al., 2003), the *direct* mutual-information-based pose estimation is used to compute the camera pose. The block diagram of the hybrid approach is shown in Fig. 4.

3. Comparative methodology

In this work, we systematically compare camera pose estimation approaches in three scenarios: firstly, we compare the performance of different pose estimation methods for single query images in the simplest scenario of using only one *reference tuple*, as shown in Fig. 1. Secondly, we increase the number of reference images and evaluate the improvement in accuracy. Thirdly, we evaluate the different approaches with large spatial uncertainties, where the reference images can be far away from the query image. The three scenarios considered for comparison are described in more detail below.

3.1. Single-reference pose estimation

The aim of using a single reference image for different pose estimation methods is to compare their performance at the most basic

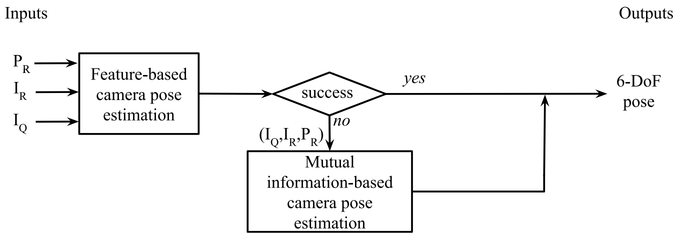

Inputs
Outputs
P_R
I_R
I_Q
Feature-based camera pose estimation
success
yes
no
(I_Q, I_R, P_R)
Mutual information-based camera pose estimation
6-DoF pose

Fig. 4. Block diagram of the hybrid approach for camera pose estimation.

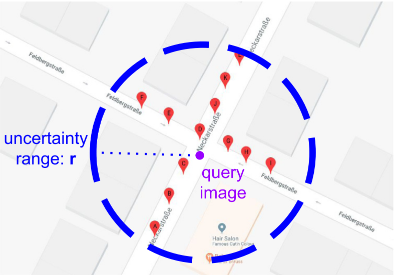


Fig. 5. Single-reference pose estimation. The actual location of the query image is marked with a purple dot, and a circle around the purple dot represents the initial uncertainty on the location of the query image. Within the uncertainty circle, one reference image is randomly selected among all possible candidates that are indicated with red markers from A to L. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

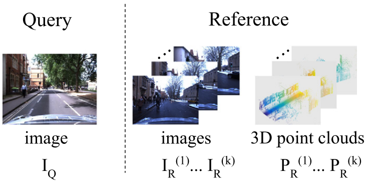


Fig. 6. Example of inputs for multi-reference case: one query image I_Q and multiple *reference tuples* $\{(I_R^{(1)}, P_R{}^{(1)}), \ldots (I_R^{(k)}, P_R{}^{(k)})\}$ which consist of k reference images and k 3D point clouds. All the reference images and 3D point clouds are defined in an unified coordinate system.

level without pre- or post-processing steps. As illustrated in Fig. 5, the experiment starts by first defining a radius r around the actual location of the query image. The radius r represents the uncertainty in the location of the query image. The reference image is randomly selected in the region within the circle. The motivation of random selection is to evaluate how the studied algorithms respond to different overlaps between query and reference images. Increasing the radius reduces the potential overlap between query and reference images, which makes pose estimation more challenging. For *direct* methods, this can be considered as different initialization. After randomly selecting one reference image within the radius, the inputs of the single-reference case are the query image I_Q and a *reference tuple* (I_R, P_R), where I_R is a single reference image and P_R is its corresponding 3D point cloud. The quality of the estimated pose is then assessed in the terms of translation error and rotation error (see Section 4.2).

3.2. Multiple-reference pose estimation

In this section we explain the case of incorporating the information obtained from multiple reference images to estimate the camera pose of a single query image. In this case, the inputs are one query image and multiple *reference tuples* which consist of k pairs of reference images and their corresponding 3D point clouds, $\{(I_R^{(1)}, P_R{}^{(1)}), \ldots (I_R^{(k)}, P_R{}^{(k)})\}$, as shown in Fig. 6. All the reference images and 3D point clouds are defined in an unified coordinate system. The aim of using multiple reference images is to leverage the additional information to improve accuracy of camera pose estimation.

In the prior art, Song et al. (2016) fuse multiple camera poses by: (1) averaging three rotation angles to compute the final rotation matrix; (2) minimizing a geometrical error term to estimate the final translation. However, 3D point clouds are not utilized in their approach, so from each reference image only a line where the camera pose of the query image should lie is obtained. In contrast, in our approach, each reference image together with a 3D point cloud are already sufficient to compute a unique 6-DoF camera pose for the query image. Therefore, we have considered 4 strategies, which can be easily adapted to different camera pose estimation methods.

1. Maximum number of matched features (*maxf*): we match the query image with all the available reference images, and select the reference image with the largest number of matched features after the feature matching stage. Then, we compute the camera pose of the query image with only the *reference tuple* that contains the selected reference image. The remaining processing steps are the same as in the camera pose estimation with a single *reference tuple*.
2. Simple average (*avg*): for each *reference tuple* in $\{(I_R^{(1)}, P_R{}^{(1)}), \ldots (I_R^{(k)}, P_R{}^{(k)})\}$, we compute an individual candidate camera pose $\mathbf{M}_{(i)} = \left[\mathbf{R}_{(i)} \quad | \quad \mathbf{t}_{(i)}\right]$ where $\mathbf{R}_{(i)}$ and $\mathbf{t}_{(i)}$ are the rotation matrix and translation vector of the ith camera pose, and $i \in \{1, \ldots, k\}$. As a result, k candidate camera poses will be obtained. Each 6-DoF camera pose consists of a rotation matrix and a translation vector. We average the k rotation matrices by firstly converting them to quaternions and then apply quaternion space interpolation (Markley et al., 2007). As a result, the final rotation matrix is obtained from the averaged quaternion representation, and the final translation vector can be computed by averaging all the translation vectors.
3. Weighted average (*wavg*): similar to *simple average*, this approach starts with k individual candidate pose estimates $\mathbf{M}_{(i)} = \left[\mathbf{R}_{(i)} \quad | \quad \mathbf{t}_{(i)}\right]$ obtained from each *reference tuple*. Then we take a weighted average of these k camera poses, and the weights $\mathbf{w}_{(i)}$ are computed according to the number of matched features between the query image and each reference image. The calculation of the final pose can be formulated as follows:

$$\mathbf{M}^* = \sum_i \mathbf{w}_{(i)}\mathbf{M}_{(i)}, \quad i \in \{1, 2 \ldots, k\} \tag{5}$$

where the rotation matrix $\mathbf{R}_{(i)}$ in $\mathbf{M}_{(i)}$ is converted to quaternions and then we compute a quaternion-weighted average (Markley et al., 2007). Each weight value is computed as follows,

$$\mathbf{w}_{(i)} = \frac{m_{(i)}}{\sum m_{(i)}}, \quad i \in \{1, 2 \ldots, k\} \tag{6}$$

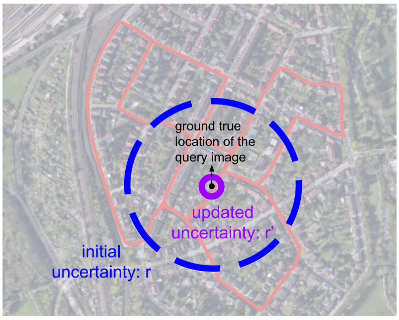


Fig. 7. Camera pose estimation with a large uncertainty. An image retrieval method is combined with a camera pose estimation method to reduce the large position uncertainty of the query image. The black dot represents the actual location of the query image, the big blue dashed circle shows the initial uncertainty and the small purple solid circle indicates the updated uncertainty in pose estimation. The red route marked in the background is one of the routes in the KITTI dataset. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

where $m_{(i)}$ is the number of the matched features between the query image I_Q and the ith reference image $I_R^{(i)}$.

4. Robust weighted average (*r-wavg*): firstly we match the query image with all the available reference images and record the numbers of their matches. If the maximum number of matched features between the query image and reference images is K, we select those reference images with at least half of the maximum matches $K/2$. The weights for individual candidate camera poses are computed as follows:

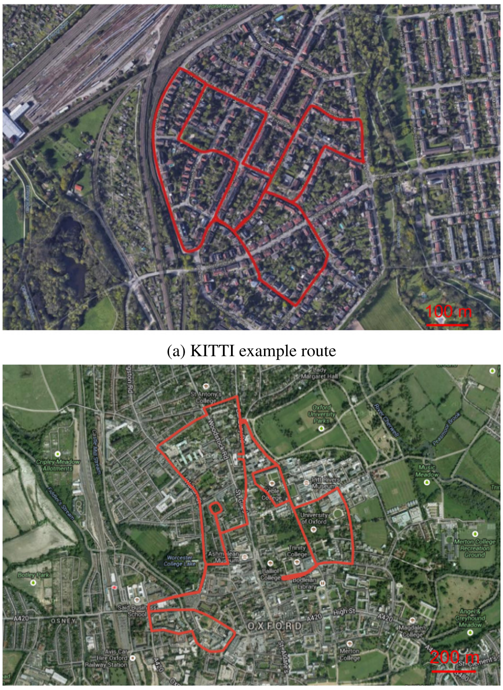
$$w(i) = \begin{cases} 0, & \text{if } m_{(i)} < \dfrac{K}{2} \\ \dfrac{m_{(i)}}{\sum m_{(i)}}, & \text{if } m_{(i)} \geq \dfrac{K}{2} \end{cases} \tag{7}$$

where K is the maximum number of matched features and it can be formulated as $K = \max\{m_{(i)}\}, i \in \{1, 2 \ldots, k\}$. In the end, we apply obtained weights to Eq. (5) to get the final camera pose.

3.3. Camera pose estimation with large uncertainties

In real-life applications, the query image may or may not have a GPS tag, and even with a GPS tag, the precision of the GPS can be poor (Linegar et al., 2016; Miura et al., 2015). Therefore, the initial uncertainty radius r of the query camera's location can be large (see Fig. 7). In the case of large uncertainties, choosing the reference image by random selection is not practical anymore, and the use of an image retrieval method becomes beneficial. Therefore, we compare the performance of the studied pose estimation methods with a large uncertainty, and evaluate how image retrieval improves their performance.

In image retrieval, methods such as Song et al. (2016), Philbin et al. (2007), Radenović et al. (2016) and Iscen et al. (2017) are used to effectively identify a few good reference images from a large reference database. In this work, we select the retrieval method (Philbin et al., 2007) which performs image retrieval from a large image set by quantizing low-level image features based on randomized trees and using an efficient spatial verification stage to re-rank the results returned from a bag-of-words model. We take up to 5 reference images with the highest scores from the retrieved ones, and then we perform single query camera pose estimation with multiple reference images.



(a) KITTI example route

(b) Oxford RobotCar route

Fig. 8. Sample routes for KITTI and Oxford RobotCar dataset with scales.

4. Experiments and results

4.1. Datasets

In this work, experiments were conducted using two public datasets: the KITTI Visual Odometry dataset (Geiger et al., 2012) and the Oxford RobotCar dataset (Maddern et al., 2017). The KITTI dataset was captured by driving around the mid-size city of Karlsruhe (Germany), in rural areas and on highways. The accurate ground truth is provided by a Velodyne laser scanner and a GPS localization system. There are 11 sequences in the KITTI dataset with ground-truth camera poses available, and we use all of them in our experiments. These sequences are summarized in Table 1. For each sequence, a 3D point cloud P_R is obtained from LIDAR, and both query image I_Q and reference image I_R are from one monochrome camera (according to the author the monochrome camera is less noisy). For illustration, one example route from the KITTI dataset is shown in Fig. 8a.

The recently released Oxford RobotCar dataset (Maddern et al., 2017) provides multiple traversals of the same route and allows a more challenging evaluation in changing weather and daylight conditions. 5 sequences of the Oxford RobotCar dataset with completely different environment conditions were selected for our experiments. The sequence route is shown in Fig. 8b and example images from 5 sequences are shown in Fig. 9. Similarly to the KITTI dataset, 3D point clouds are obtained from LIDAR. The reported GPS information is treated as the ground-truth for the camera location.

The Oxford RobotCar dataset includes images captured by a Bumblebee XB3 (1280 × 960 × 3, 16 Hz). In our experiment, we use the left image from the Bumblebee XB3 and, for efficiency, we reduced the number of images in each sequence by taking 1 out of every 10 images. Also we removed the beginning and ending frames of each sequence

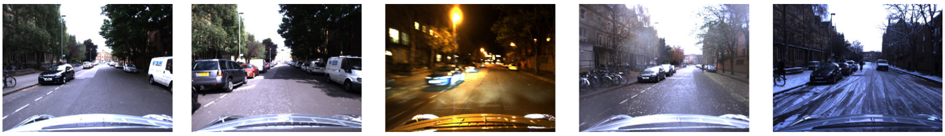

(a) overcast (b) sun (c) night (d) rain (e) snow

Fig. 9. Appearance differences among the 5 sequences in the Oxford RobotCar dataset (the images are roughly from the same location).

Table 1
Overview of the 11 sequences in the KITTI dataset (Geiger et al., 2012).

Id	# images	Tag	Total length (km)	Mean distance between consecutive images (m)
00	4541	Urban	3.7	0.8
01	1101	Highway	2.5	2.2
02	4661	Urban	5.1	1.1
03	801	Urban	0.6	0.7
04	271	Urban	0.4	1.5
05	2761	Urban	2.2	0.8
06	1101	Urban	1.2	1.1
07	1101	Urban	0.7	0.6
08	4071	Urban	3.2	0.8
09	1591	Urban	1.7	1.1
10	1201	Urban	0.9	0.8

Table 2
Overview of 5 sequences with different environmental conditions in Oxford RobotCar dataset (Maddern et al., 2017).

Id	# images	Tag	Total length (km)	Mean distance between consequent images (m)
00	1916	Overcast	6.3	3.3
01	2873	Sun	8.6	3.0
02	2931	Night	9.1	3.1
03	2614	Rain	8.8	3.4
04	3019	Snow	8.7	2.9

where the car is usually parked, producing multiple instances of the same image. The resulting 5 sequences from Oxford RobotCar dataset are summarized in Table 2. For the Oxford RobotCar dataset, the query I_Q and reference I_R images are taken from different traversals of the route, and therefore give a much more demanding assessment of pose estimation performance in realistic conditions.

There are two main reasons why we used these specific datasets. One is the availability of ground truth from commercial-level Inertial and GPS navigation system. For example, KITTI dataset uses OXTS RT 3003 (Oxford-Technical-Solutions-Ltd, 2019), and Oxford RobotCar dataset uses NovAtel SPAN-CPT ALIGN (NovAtel-Inc., 2019). This type of ground truth information is very limited in other existing datasets. The other reason is that Oxford RobotCar dataset consist of the multiple traversals of the same route under changing weather and daylight conditions. However, since the both datasets are acquired by sensors on a car the main application field of our results is self-driving cars. This indicates certain limitations in the images, such as the small variation in viewpoint between consecutive frames.

4.2. Performance measures

We use translation error, maximum orientation error and the success rate of each method to compare the performance of the different approaches:

1. The translation error is the absolute translation between the ground-truth location and the estimated location of the query image.
2. Based on the rotation matrix between the ground-truth camera pose and the estimated camera pose of the query image, we convert the rotation matrix into 3 Euler angles. Then the maximum absolute Euler angle is used as the maximum orientation error.
3. The studied methods can fail to yield a camera pose estimate under some circumstances, for instance when there are not enough feature matches between the query and reference images in the *indirect* approach, or when grid search fails to converge in *direct* approaches. In this work, we define the *success rate* as the percentage of the processed images for which the estimated poses are within 10 m from ground truth, and this threshold is picked from the prior art (Pascoe et al., 2017).

4.3. Experiments with single reference image

In this section, we perform 12 sets of experiments for both KITTI and Oxford RobotCar datasets. Each set of experiments comprises hundreds of estimates for a pose estimation method at an uncertainty radius. The goal of these experiments was to compare the performance of different pose estimation methods under the single reference scenario, as described in Section 3.1. For the experiments, the uncertainty radius r was varied between 10 to 25 m. Since most of the photos are taken by a front-looking camera mounted in a car in the streets of an urban area, these search ranges were selected so that the reference and query images would have some overlap but not being too close to each other. The mean distance between two consequent images are from 0.7 to 3.4 m in the two evaluated datasets.

The experiments with the KITTI dataset tested the performance of different camera pose estimation methods under "ideal conditions", *i.e.* same time of the day, lighting and weather condition. For the KITTI dataset, all the 11 sequences listed in Table 1 have different routes. For this reason, each sequence was processed individually so that the query image and the reference images come from the same drive. In order to separate the query and reference images, we randomly selected 10% of the images in one sequence for queries, and the rest of images from the same sequence were used as references.

The experiments with the Oxford RobotCar dataset tested the performance of camera pose estimation methods in challenging conditions since the query and reference data capture large variation in appearance and structure of a dynamic city environment over long periods of time. For the Oxford RobotCar dataset presented in Table 2, each one of the 5 route traversals corresponds to different environmental conditions on the same route. The sequences were processed jointly in order to allow the query and reference images to come from the different sequences. For example, when the summer sunny sequence (01 in Table 2) was used for the reference images, the winter snow sequence (04 in Table 2) was used for the queries.

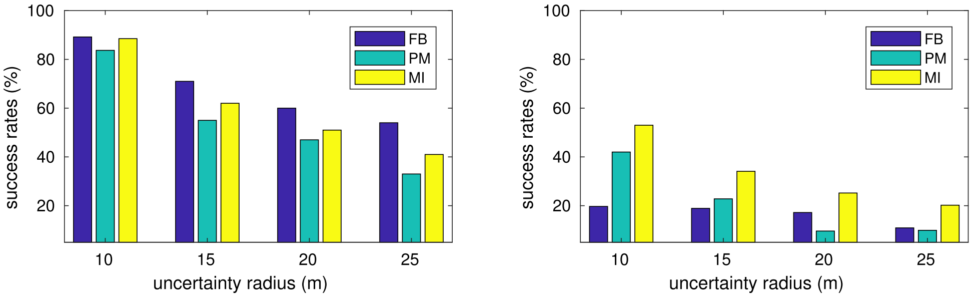
Table 3 summarizes the translation and orientation errors for the studied methods (FB, PB and MI) in the single-reference scenario. For a fair comparison of the performance measures, we decided to use only those images for which all methods are able to provide a pose estimate (regardless of accuracy). From Table 3a and b we observe the following:

1. By looking into each column, we find that as long as the feature-based approach is able to estimate the camera pose, its estimates have smaller translation errors than the other two methods in

Table 3
Translation error (in meters) and **maximum orientation error** (in degrees) using a single reference image. For the KITTI dataset, 454 images (random 10% of the whole sequence) in sequence 00 are used as queries, and the rest as the reference images. For the Oxford RobotCar dataset, summer sequence (01) is used as the reference and 302 images (random 10%) from the winter sequence (04) are used as the query images. The second row shows the number of images for which all methods are able to provide a pose estimate regardless of accuracy. The third row shows the percentage value. All the translation and orientation results are reported in median values.

(a) KITTI sequence: translation error (m)					(b) Oxford sequence: translation error (m)				
Uncertainty radius (m)	10	15	20	25	Uncertainty radius (m)	10	15	20	25
#images	406	328	282	259	#images	67	60	53	38
	(89%)	(72%)	(62%)	(57%)		(22%)	(20%)	(18%)	(13%)
FB (Kim et al., 2014)	**0.13**	**0.40**	**0.48**	**0.30**	FB (Kim et al., 2014)	**2.77**	**2.48**	**2.40**	**2.91**
PM (Tykkälä et al., 2013)	1.44*	6.66*	7.77*	14.85*	PM (Tykkälä et al., 2013)	10.44*	16.23*	20.09*	26.32*
MI (Pascoe et al., 2017)	1.56*	5.41*	6.15*	10.26*	MI (Pascoe et al., 2017)	8.71*	13.36*	16.27*	14.94*
(c) KITTI sequence: max orientation error (degree)					(d) Oxford sequence: max orientation error (degree)				
Uncertainty radius (m)	10	15	20	25	Uncertainty radius (m)	10	15	20	25
#images	406	328	282	259	#images	67	60	53	38
	(89%)	(72%)	(62%)	(57%)		(22%)	(20%)	(18%)	(13%)
FB (Kim et al., 2014)	1.76	3.83	5.42	3.33	FB (Kim et al., 2014)	**3.44**	**3.79**	2.72	3.25
PM (Tykkälä et al., 2013)	**1.07***	2.40*	**3.37***	3.12*	PM (Tykkälä et al., 2013)	3.48*	5.82	2.64	**1.88**
MI (Pascoe et al., 2017)	**1.07***	**2.30***	3.45*	**2.70***	MI (Pascoe et al., 2017)	6.16	4.00	**2.42**	1.93*

*Indicates a statistically significant difference at the $p < 0.05$ level computed with the Wilcoxon signed rank test (Gibbons and Chakraborti, 2011) against the Feature-based method (FB).



(a) KITTI

(b) Oxford RobotCar

Fig. 10. Success rate comparison for three strategies with single reference image at different uncertainty ranges in two public datasets. (a): in the experiments with the KITTI sequence 00, a random 10% of the images are used as query image and the rest are used as references. (b): in the experiments with two sequences in Oxford RobotCar sequences, summer sequence (01) was used for the references and the snow sequence (04) was used for queries. Failure threshold was set to 10 m.

both the KITTI and Oxford RobotCar datasets. This result indicates that the feature-based approach is more accurate in pose estimation in both ideal environment conditions (KITTI dataset) and realistic environment conditions (Oxford RobotCar dataset) with random reference image selection.

2. By looking into each row, we find that the translation errors of both photometric-based and mutual-information-based approach increase with the increasing **uncertainty radius**, but the translation error of the feature-based approach does not vary much. This suggests that both the photometric and mutual-information-based approaches are more sensitive to the initialization.

Table 3c and d compare the orientation errors. Among the studied methods, the differences in their orientation errors are small. In other words, all these methods perform similarly in terms of orientation error for both KITTI and Oxford RobotCar datasets. The reason for these results might be that all the images are taken by a front-looking camera mounted on a car driving along the street, so the query images and the reference images may share similar viewpoints. Fig. 10 plots the *success rates* (see definition in Section 4.2) for the studied three strategies with a single *reference* at different uncertainty ranges. Fig. 10 shows the following:

1. The feature-based approach has higher success rate than the other two approaches in the KITTI dataset; however, the feature-based approach has the lowest success rate among all three approaches in the Oxford RobotCar dataset. The mutual-information-based approach has the highest success rate in Oxford RobotCar dataset. This suggests that the success rate of the feature-based approach is greatly influenced by the environmental conditions between the query and reference images. On the other hand, the mutual-information-based approach is the most robust in terms of the success rate under different environmental conditions.
2. When analyzing the same pose estimation method for different uncertainty radii, the *success rates* of all approaches decrease with the increase of the uncertainty radius.

Pascoe et al. (2017) claim that the mutual-information-based SLAM approach has higher success rate than state-of-the-art feature-based SLAM approaches (Mur-Artal et al., 2015). Our experiments in Fig. 10b lead to the same conclusion in the problem of 6-DoF camera pose estimation using single reference image and 3D point cloud. Interestingly enough, our experiments in Table 3 suggest that the feature-based approach can be more accurate as long as it is able to compute the camera pose.

The observations presented above lead us to use the hybrid (HY) method for pose estimation. Recall however that, for the results presented in Table 3, we selected images for which all the methods yield a pose estimate. As a result, the performance of the hybrid method (HY) in this setting is equivalent to the feature-based method (FB) since the photometric-based branch of the HY approach works only when the FB method fails. For this reason, we only include the HY approach in the large uncertainty scenario presented in Section 4.5.

4.4. Experiments with multiple reference images

In this experiment, we evaluated the performance of different methods in the multi-reference setting for the both KITTI and Oxford RobotCar datasets. The goal was to evaluate efficient ways to incorporate the information obtained from multiple reference images to improve the camera pose estimation.

Similarly to the single reference case of previous section, we consider the reference images within the uncertainty radius r around the actual location of the query image, and then randomly selected multiple *reference tuples*. Subsequently, we evaluated the 4 different methods to fuse camera poses from multiple *reference tuples*: maximum number of matched features (*maxf*), simple average (*avg*), weighted average (*wavg*) and the robust weighted average (*r-wavg*). The number of reference images was varied from one to five.

The results for different multi-reference pose estimation methods in the KITTI dataset are shown in Table 4. Fig. 11 compares the *success rates* for different camera pose estimation methods with multiple reference images using the *robust weighted average (r-wavg)* method in the both KITTI and Oxford RobotCar datasets. The *r-wavg* method was used in that figure since it yielded the best overall performance for all the pose estimation methods.

Table 4 summarizes the results for the experiments with multiple reference images. The results show that fusing the poses from multiple references improves the performance of the camera pose estimation results, and *robust weighted average* (*r-wavg*) outperforms the other approaches, especially with the increased number of reference images. Fig. 11 compares the *success rates* of the different approaches with multiple reference images using *robust weighted average* method in the both KITTI and Oxford RobotCar datasets. Fig. 11 tells us two things:

1. The success rates of each method show that the *success rate* increases with the increase of the number of reference images.
2. The three bars at each plot show that the feature-based approach has the highest success rate among different approaches in the KITTI dataset, but has the lowest success rate in the Oxford RobotCar dataset. In contrast, the mutual-information-based approach has the highest success rate in that dataset. In other words, mutual information is more robust than the two other approaches under changing environmental conditions. This finding is consistent with our results in the single reference scenario.

In the literature, camera pose estimation usually requires geometry verification (Sattler et al., 2016) which is very effective but requires extra computation. Interestingly enough, our results show that the *robust weighted average* method is a light approach and can be easily adapted with any pose estimation method with good results.

4.5. Experiments at large uncertainty

Based on the empirical results in Section 4.3, we evaluated a hybrid approach that leverages the advantages of both the feature-based and the mutual-information-based approaches. In this section, we tested these 4 camera pose estimation methods (feature-based, photometric-based, mutual-information-based, and hybrid approaches) with five reference images, under the large uncertainty condition.

In Section 3.3, we described the experimental setting for camera pose estimation under large location uncertainty. In the extreme case, no prior information on the location is available and the query image must be matched to the whole reference database. As a result, an image retrieval method is applied to find suitable reference images (Philbin et al., 2007). Among all retrieved reference images, up to 5 images with the highest scores are stored for further processing. In our experiments we restricted the uncertainty radius to 200 m for the KITTI and 50 m for the Oxford RobotCar dataset, and adopted the multi-references (up to 5 most similar reference images) pose-estimation approach to improve robustness of all the investigated methods. We conducted experiments in all the sequences of the KITTI dataset. In the Oxford RobotCar dataset, we performed a set of experiments where one sequence is used for the references and another sequence is used for the queries.

The results for the KITTI and Oxford RobotCar datasets are shown in Tables 5 and 6 respectively. In this two tables, we tag a pose estimate as a *failure* when the translation error is above 10 m. By looking at the *success rates* in Table 5, we see that the hybrid and feature-based approaches outperform other methods in cases where the query and reference images have been captured at similar imaging conditions (KITTI dataset). The hybrid approach performs similarly as the feature-based approach, which indicates that the evaluated hybrid method can retain good properties of the feature-based method. For the sequence 01, the hybrid method is superior. The plausible explanation for this is that the sequence 01 is captured from a highway (see Table 1) where there are less reliable features to be found than in urban scenes. In urban scenes, the hybrid and feature-based methods provide practically the same accuracy. Table 6 shows a confusion matrix summarizing the results in the Oxford RobotCar dataset. For that table, we repeated the experiments by using one sequence as reference and another one as query (a total of 5 × 5 different combinations). Therefore, in addition to a large spatial displacement, query and reference images have been acquired at very different imaging conditions. From that table, we conclude the following:

1. The mutual-information-based approach is more robust than the feature-based or photometric-based approaches, which is consistent with the findings in the single reference and multi-reference scenarios.
2. The hybrid approach outperforms all other approaches in success rate when the query and reference images have very different imaging conditions. This confirms that the hybrid method leverages complementary properties of the feature-based and mutual-information-based methods.

The results on the diagonal of Table 6 are consistent with previous experiments in the KITTI dataset in Table 5, i.e. in the ideal case when query and reference images come from the same sequence and imaging conditions. In this case, feature-based and our hybrid method outperform the other approaches. A remarkable result in Table 6 is that, even in the worst case scenario, the lowest success rate of the hybrid method is 13.2%. Recent results in the same dataset in similar conditions have reported *success rates* as low as 0% using SLAM (Pascoe et al., 2017). Notice that the experimental settings in that work (Pascoe et al., 2017) are different from ours, but this helps understanding the difficulty of pose estimation problem under real conditions.

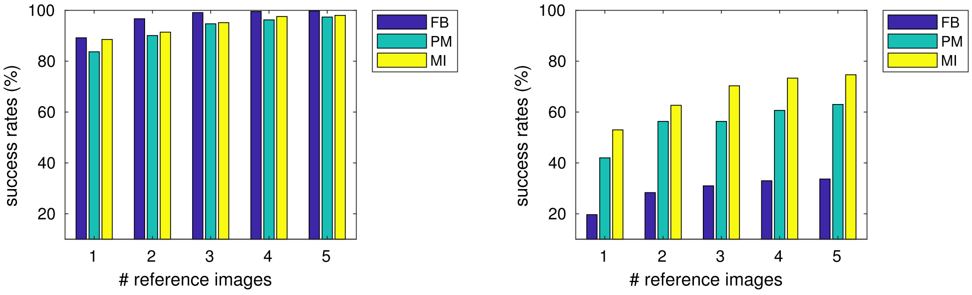
5. Conclusion

We performed systematic and extensive comparisons of three different strategies for 6-DoF camera pose estimation using reference images and 3D point clouds: an *indirect* feature-based approach, a *direct* photometric-based approach and a *direct* mutual-information-based approach. In our experiments the feature-based approach was more accurate than both the photometric-based and mutual-information-based approaches when as few as 4 consistent correspondent points were found between query and reference images. The mutual-information-based approach was more robust than the feature-based and photometric-based approaches which means that it can provide an estimate even in the cases when the other methods fail. As expected, the robustness and accuracy of all methods improved when multiple reference images were available. In the multi-reference scenario, the *robust weighted average* method outperformed other fusing methods for the estimation of the pose from multiple candidates. Based on the strong

Table 4
Performance in multi-reference pose estimation in the KITTI sequence 00. 10% images (454) from this sequence are used as query image and the rest are used as references. The uncertainty radius is $r = 10$ m. The reported results are computed from those images for which all methods are able to provide a pose estimate.

#reference images	1		2		3		4		5	
	Median (m)	Median (deg)	Median (m)	Median (deg)	Median (m)	Median (deg)	Median (m)	Median (deg)	Median (m)	Median (deg)
Feature-based (FB)										
avg	0.13	1.76	0.22*	2.07*	0.25*	2.20*	0.21*	1.80*	0.19*	1.61*
wavg	0.13	1.76	0.15*	**1.67***	0.15*	1.78	0.10*	1.22*	0.09*	1.11*
maxf	0.13	1.76	**0.11**	1.82	**0.09**	1.79*	0.06	1.21*	0.05*	1.03*
r-wavg	**0.13**	**1.76**	0.12	1.70	0.10	**1.59**	**0.06**	**1.13**	**0.04**	**0.93**
Photometric (PM)										
vg	1.44	1.07	2.29*	1.26*	2.15*	1.38*	2.08*	1.22*	1.90*	1.05*
avg	1.44	1.07	1.67*	1.00*	1.52	1.07*	1.28*	0.79*	1.12*	0.69*
axf	1.44	1.07	**1.34**	1.01	1.22	1.01*	1.19*	0.72*	1.07	0.66*
r-wavg	**1.44**	**1.07**	1.35	**0.95**	**1.21**	**0.86**	**1.12**	**0.68**	**0.99**	**0.58**
Mutual Information (MI)										
avg	1.56	1.07	1.65	1.24*	1.80*	1.43*	1.68*	1.23*	1.60*	1.12*
wavg	1.56	1.07	1.44*	1.10	1.37	1.20	1.16	0.77*	1.17	0.77*
maxf	1.56	1.07	**1.36***	1.07	1.29*	1.02*	1.16*	0.79*	1.12*	0.68*
r-wavg	**1.56**	**1.07**	1.38	**0.98**	**1.25**	**0.94**	**1.09**	**0.68**	**1.03**	**0.62**

*Indicates a statistically significant difference at the $p < 0.05$ level computed with the Wilcoxon signed rank test (Gibbons and Chakraborti, 2011) against the robust weighted average method (r-wavg).



(a) KITTI (b) Oxford RobotCar

Fig. 11. **Success rates** comparison for the studied methods with multiple reference images and *robust weighted average* method in two datasets. The failure threshold was set to 10 m.

empirical results and inspired by the complementary properties of the feature-based and mutual-information-based approaches, we evaluated a computationally cheap and easy-to-adapt hybrid approach that combines these two methods. In all experiments, the hybrid method was on par or superior to the single methods. This is particularly so in challenging scenarios such as the Oxford RobotCar dataset, where the hybrid approach outperforms feature-based and mutual-information-based approaches by an average increase in success rate of 9.4% and 8.7%, respectively.

In our experiments with multiple reference images (Section 4.4), we tested different fusion methods to compute the camera pose. The speed of the photometric and mutual-information methods could be greatly improved by utilizing GPU or thread programming. Based on the experimental results, we empirically fixed the number of the reference images to be 5 in the large uncertainty case. An interesting question to be addressed in the future work is to investigate the optimal number of images needed to achieve a certain accuracy and to compare different image retrieval approaches.

Acknowledgement

This work is supported by Academy of Finland under Grant No. 314180 and Czech Science Foundation under Grant GA18-05360S.

Appendix A. Indirect feature-based pose estimation

This appendix presents the detailed description of the four stages of the *indirect* feature-based pose estimation method presented in Section 2.1.

A.1. Feature detection and description

The first step of the system is to detect and extract features of salient locations in the query and reference images. Specifically, a feature detector is used for finding the salient points of an image, and a feature descriptor is used to describe the neighborhood surrounding that salient point.

Feature detectors can extract different types of image structures, e.g. corners (Rosten and Drummond, 2006; Mikolajczyk and Schmid, 2004), blobs (Lowe, 1999; Bay et al., 2006; Kadir and Brady, 2001) or regions (Matas et al., 2004; Tuytelaars and Van Gool, 2000, 2004; Mori et al., 2004). For reference purposes, a summary of invariance properties and performance analysis for some feature detectors are shown in Table A.7. In turn, feature descriptors can be divided into following categories: local binary descriptors (Ojala et al., 2002; Guo et al., 2010; Zhao and Pietikainen, 2007; Froba and Ernst, 2004; Calonder et al., 2010; Rublee et al., 2011; Leutenegger et al., 2011; Alahi et al., 2012), spectral descriptors (Lowe, 1999; Lienhart and Maydt, 2002; Bay et al., 2006; Dalal and Triggs, 2005; Tola et al., 2010; Ambai and Yoshida, 2011), basis space descriptors (Zahn and Roskies, 1972; Csurka et al., 2004), polygon shape descriptors (Matas et al., 2004; Belongie et al., 2001), 3D and volumetric descriptors (Klaser et al., 2008; Scovanner et al., 2007). In the literature, many feature

Table 5
Camera pose estimation results in a large uncertainty for all 11 KITTI sequences. The uncertainty radius was set to be 200 m and 5 best retrieved reference images were used for camera pose estimation. The failure threshold was set to 10 m. Note that the first three evaluated methods (Tykkälä et al., 2013; Pascoe et al., 2017) were originally designed for visual SLAM, but we modified the algorithms for the pose estimation problem.

#sequence ID	00			01			02			03		
	%	Median (m)	Median (deg)	%	Median (m)	Median (deg)	%	Median (m)	Me-dian(deg)	%	Median (m)	Median (deg)
FB (Kim et al., 2014)	**99.8**	0.031	0.676	84.5	0.494	0.567	**99.8**	0.025	0.415	**100**	0.015	0.370
PM (Tykkälä et al., 2013)	98.2	0.603	0.423	76.4	1.208	0.343	92.9	0.550	0.324	98.8	0.342	0.279
MI (Pascoe et al., 2017)	97.8	0.633	0.415	60.0	0.980	0.353	97.6	0.475	0.327	98.8	0.270	0.223
HY (Combine FB and MI)	**99.8**	0.031	0.676	**89.1**	0.505	0.562	**99.8**	0.025	0.415	**100**	0.015	0.370
#sequence ID	04			05			06			07		
	%	Median (m)	Median (deg)	%	Median (m)	Median (deg)	%	Median (m)	Median (deg)	%	Median (m)	Median (deg)
FB (Kim et al., 2014)	**100**	0.028	0.132	**100**	0.022	0.472	**100**	0.029	0.421	**100**	0.018	0.326
PM (Tykkälä et al., 2013)	96.3	0.783	0.222	97.8	0.514	0.360	98.2	0.382	0.308	97.3	0.505	0.336
MI (Pascoe et al., 2017)	**100**	0.495	0.177	97.1	0.537	0.352	96.4	0.551	0.332	98.2	0.500	0.319
HY (Combine FB and MI)	**100**	0.028	0.132	**100**	0.022	0.472	**100**	0.029	0.421	**100**	0.018	0.326
#sequence ID	08			09			10					
	%	Median (m)	Median (deg)	%	Median (m)	Median (deg)	%	Median (m)	Median (deg)			
FB (Kim et al., 2014)	**100**	0.018	0.383	99.4	0.019	0.356	**100**	0.019	0.420			
PM (Tykkälä et al., 2013)	97.3	0.499	0.329	95.0	0.548	0.321	94.2	0.634	0.355			
MI (Pascoe et al., 2017)	95.3	0.518	0.341	93.7	0.400	0.350	91.7	0.780	0.343			
HY (Combine FB and MI)	**100**	0.018	0.383	**100**	0.019	0.368	**100**	0.019	0.420			

Table 6
Large uncertainty pose estimation results for the 5 different sequences in Oxford RobotCar dataset. The uncertainty radius was set to 50 m and 5 best retrieved reference images are used for camera pose estimation. The failure threshold was set to 10 m. Note that the first three evaluated methods (Tykkälä et al., 2013; Pascoe et al., 2017) were originally designed for visual SLAM, but we modified the algorithms for the pose estimation problem.

		Overcast			Sun			Night			Rain			Snow		
		%	Median (m)	Median (deg)	%	Median (m)	Median (deg)	%	Median (m)	Median (deg)	%	Median (m)	Median (deg)	%	Median (m)	Median (deg)
Overcast	FB (Kim et al., 2014)	98.4	0.111	0.791	31.1	3.280	3.015	5.6	6.281	4.043	27.1	3.355	4.106	16.7	1.407	6.941
	PM (Tykkälä et al., 2013)	98.4	1.599	0.578	6.2	7.751	6.006	7.3	7.347	10.397	6.7	6.534	3.319	5.3	7.718	9.171
	MI (Pascoe et al., 2017)	99.0	1.551	0.716	16.3	4.648	3.991	15.5	7.479	8.625	12.0	7.363	11.716	18.2	6.902	6.556
	HY (Combine FB and MI)	**100.0**	0.112	0.788	**40.1**	3.398	3.103	**21.0**	6.966	4.486	**35.1**	3.828	5.029	**31.1**	4.251	6.687
Sun	FB (Kim et al., 2014)	33.3	2.604	1.993	98.3	0.121	0.706	2.9	4.642	9.733	12.1	2.275	3.963	15.2	2.877	3.527
	PM (Tykkälä et al., 2013)	10.7	6.242	2.135	96.2	1.919	0.585	9.2	7.412	13.885	8.2	6.968	9.364	5.0	7.080	4.104
	MI (Pascoe et al., 2017)	16.7	5.102	3.722	96.2	1.685	0.569	17.6	5.859	8.779	11.7	7.470	6.825	15.6	7.793	4.937
	HY (Combine FB and MI)	**40.0**	3.010	2.342	**99.7**	0.122	0.715	**20.1**	5.350	8.722	**21.8**	5.324	4.890	**26.2**	4.514	3.712
Night	FB (Kim et al., 2014)	4.9	3.332	2.647	1.8	5.337	7.200	89.4	0.217	0.744	1.2	2.879	4.171	2.3	5.788	8.191
	PM (Tykkälä et al., 2013)	5.9	8.255	12.956	3.2	8.437	3.251	90.8	2.303	0.543	4.3	8.725	8.275	2.3	6.973	4.625
	MI (Pascoe et al., 2017)	8.8	7.189	8.960	11.9	6.732	9.110	94.5	2.126	0.554	12.0	7.776	6.299	13.7	8.199	6.209
	HY (Combine FB and MI)	**13.7**	5.983	3.966	**13.3**	6.424	7.745	**96.9**	0.233	0.811	**13.2**	6.996	5.593	**15.0**	7.257	7.406
Rain	FB (Kim et al., 2014)	31.6	3.251	2.536	13.2	2.264	4.631	3.7	2.006	1.504	96.9	0.192	0.764	17.2	3.289	3.619
	PM (Tykkälä et al., 2013)	13.5	6.959	2.313	13.6	6.913	3.210	9.2	7.050	6.144	96.2	2.336	0.578	9.3	6.436	4.910
	MI (Pascoe et al., 2017)	9.4	6.847	9.182	13.9	6.631	5.906	11.7	7.731	9.125	95.0	1.915	1.067	17.5	6.135	5.652
	HY (Combine FB and MI)	**37.4**	3.295	2.908	**24.4**	3.724	5.779	**14.7**	6.447	7.147	**99.2**	0.200	0.773	**30.8**	4.286	4.577
Snow	FB (Kim et al., 2014)	9.9	2.521	3.625	10.1	2.310	7.945	2.2	5.733	13.984	10.8	2.260	4.811	97.7	0.145	0.834
	PM (Tykkälä et al., 2013)	5.4	7.192	33.412	2.4	8.353	20.601	4.8	7.529	5.721	3.6	5.899	14.798	95.7	2.026	0.553
	MI (Pascoe et al., 2017)	12.6	8.000	7.760	12.5	7.073	4.269	13.6	7.559	8.608	8.0	6.417	9.510	95.4	2.012	0.734
	HY (Combine FB and MI)	**19.8**	4.534	4.230	**20.9**	5.343	5.731	**15.0**	6.993	8.608	**18.4**	3.727	7.399	**100.0**	0.149	0.804

descriptors, such as SURF (Bay et al., 2006), BRISK (Leutenegger et al., 2011) and others, provide their own detector method along with the descriptor method. DoG (Lowe, 1999) and SURF (Bay et al., 2006) detectors were designed for efficiency and the other properties are slightly compromised. However, for most applications they are still more than sufficient (Tuytelaars et al., 2008).

A summary of the invariance properties of the detectors is in Table A.7. In two public datasets used in this work we use images taken by a front-looking camera mounted in the car, so those images have similar viewpoints which is along the road. In this work we have utilized SURF (Bay et al., 2006) for both feature detection and description due to its invariance properties, performance, and widespread use in

Table A.7
Invariance properties of feature detectors (Tuytelaars et al., 2008).

F-detector	Invariance		
	Rotation	Scale	Affine
Harris	✓		
Hessian	✓		
SUSAN	✓		
Harris–Laplace	✓	✓	
Hessian–Laplace	✓	✓	
DoG	✓	✓	
Salient regions	✓	✓	✓
SIFT	✓	✓	
MSER	✓	✓	✓
SURF	✓	✓	

multiple applications. Another reason is that our evaluated *indirect* method (Kim et al., 2014) also uses SURF (Bay et al., 2006), and we would like to implement it in the same way.

A.2. Feature matching

Based on the previously computed feature descriptors, the aim of feature matching is finding 2D-to −2D correspondences between feature points in the query and reference image.

The popular approaches for feature matching are *exhaustive search*, *hashing* (Strecha et al., 2012), and *nearest neighbor techniques* (Friedman et al., 1977; Lowe, 2004; Muja and Lowe, 2009). *Exhaustive search* is achieved by minimizing pairwise distance measures between the feature vectors of the reference and query image. The *hashing* approach reduces the size of the descriptors by finding a more compact representation, e.g. binary strings (Strecha et al., 2012). In *nearest neighbor techniques*, KD-trees (Friedman et al., 1977) and their variants (Lowe, 2004; Muja and Lowe, 2009) are commonly used to quickly find approximate nearest neighbors in a relatively low-dimensional real-valued space. The algorithm works by recursively partitioning the set of training instances based on a median value of a chosen attribute (Friedman et al., 1977).

We use the exhaustive search approach and adopt a minimum Euclidean distance on the descriptor vector. For each feature point in one image, we find the nearest neighbor as its corresponding feature point in the other image. Besides, we reject some ambiguous matches by comparing the distance of the closest neighbor to that of the second-closest neighbor. In other words, correct matches need to have the closest neighbor significantly closer than the second closest match to achieve reliable matching (Lowe, 2004). The output of the feature matching steps are a set C of n 2D-to −2D correspondences between the query image I_Q and reference image I_R:

$$C = \{(\mathbf{p}_Q^{(1)}, \mathbf{p}_R^{(1)}), (\mathbf{p}_Q^{(2)}, \mathbf{p}_R^{(2)}), \dots, (\mathbf{p}_Q^{(n)}, \mathbf{p}_R^{(n)})\} \tag{A.1}$$

where $\mathbf{p}_Q^{(i)} = [u_Q^{(i)}, v_Q^{(i)}]^T$ and $\mathbf{p}_R^{(i)} = [u_R^{(i)}, v_R^{(i)}]^T$ are the ith 2D feature locations on reference and query images, respectively.

A.3. 2D-3D correspondences

The 2D-3D correspondences between the query image and the 3D point cloud are established by using the set C of 2D-2D matches and the point cloud P_R. Since the point cloud P_R and the reference image I_R are pre-registered and defined in the same world coordinate system, with the 2D-2D matched features, we could indirectly link the 2D-3D correspondences as illustrated in Fig. A.12.

However, if the matched 2D features at the reference image do not have associated 3D points from the pre-registered point cloud, we need to compute the 2D-3D correspondences by following steps: (1) project 3D point cloud onto the reference image, (2) compute the depth of the feature points, (3) find the corresponding 3D coordinates.

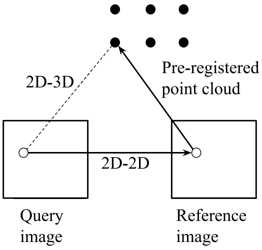


Fig. A.12. Build 2D-3D correspondences through the 2D-2D matched features and the pre-registered point cloud.

Firstly, we project the 3D point cloud $P_R = [\mathbf{P}_R^{(1)}, \mathbf{P}_R^{(2)}, \dots, \mathbf{P}_R^{(m)}]$ onto the reference image plane, and get a set of 2D projections $p = [\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \dots, \mathbf{p}^{(m)}]$, as shown in Fig. A.13. For the ith 3D point, $\mathbf{P}_R^{(i)} = [x^{(i)}, y^{(i)}, z^{(i)}, 1]^T$, we generate a 2D projection $\mathbf{p}^{(i)} = [u^{(i)}, v^{(i)}]^T$ on the reference image plane by:

$$\mathbf{p}^{(i)} = \mathbf{K} \quad \mathbf{M} \quad \mathbf{P}_R^{(i)} \tag{A.2}$$

where $\mathbf{M}$ is the world to camera transformation matrix and $\mathbf{K}$ is the intrinsic matrix of the reference image. $\mathbf{M}$ and $\mathbf{K}$ can be represented by (A.3) and (A.4):

$$\mathbf{M} = \begin{bmatrix} \mathbf{R} & | & \mathbf{t} \end{bmatrix} \tag{A.3}$$

where $\mathbf{R}$ is a 3 × 3 rotation matrix, and $\mathbf{t}$ is a 3 × 1 translation vector.

$$\mathbf{K} = \begin{bmatrix} f_x & \gamma & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.4}$$

where f_x and f_y are focal lengths (in pixels) along the x and y axis directions; γ represents the skew coefficient between x and y axis and it is often 0; u_0 and v_0 represents the principal point which would ideally be in the center of the image. In the experiments of this paper, we assume the query image and the reference images share the camera intrinsic matrix, because the images from each dataset are captured with the same camera device.

Secondly, we use nearest-neighbor search (Friedman et al., 1977) to find the closest point among 2D projections p for each 2D feature point in C at the reference image. In particular, the jth feature point $\mathbf{p}_R^{(j)}$ in the reference image is associated to the kth point of the 2D projection set p by:

$$k = NN(\mathbf{p}_R^{(j)}, p), \quad k \in \{1, 2 \dots, m\} \tag{A.5}$$

Finally, we find the 3D coordinates for each 2D feature point. In particular, the kth depth value corresponding to $\mathbf{p}^{(k)}$, namely $z^{(k)}$, is then used to find the 3D coordinates in the reference image frame corresponding to $\mathbf{p}_R^{(j)}$ as:

$$\mathbf{P}^{(j)} = \begin{bmatrix} \mathbf{K}^{-1}\mathbf{p}_R^{(j)} z^{(k)} \\ z^{(k)} \end{bmatrix} \tag{A.6}$$

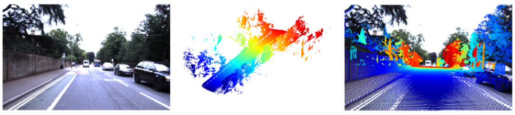
As a result, the final 2D-to −3D correspondences can be expressed as:

$$\hat{C} = \{(\mathbf{p}_Q^{(1)}, \mathbf{P}^{(1)}), (\mathbf{p}_Q^{(2)}, \mathbf{P}^{(2)}) \dots, (\mathbf{p}_Q^{(n)}, \mathbf{P}^{(n)})\} \tag{A.7}$$

where $\mathbf{p}_Q^{(i)}$ is the ith 2D feature location in the query image, and $\mathbf{P}^{(i)}$ is the ith corresponding 3D location in the reference image coordinate.

A.4. Perspective-n-point and RANSAC

The set of 2D-3D correspondences $\hat{C}$ establishes one-to-one correspondences between 2D points in the query image frame $\mathbf{p}_Q^{(j)}$, and 3D



(a) reference image (b) 3D point cloud (c) projections

Fig. A.13. An example of projecting the 3D point cloud into the reference image.

points in the reference image frame $\mathbf{P}^{(j)}$, for $j = 1, 2, \ldots, n$. The last step is to apply the Perspective-n-Point solver (Gao et al., 2003) to compute the relative 6-DoF camera pose $\mathbf{M}$ between the query image and the reference image. For this purpose, two approaches are combined to solve the problem: the algebraic approach and the geometric approach. In the algebraic approach, we use Wu's zero decomposition method (Wen-Tsun, 1986) to find a complete triangular decomposition of a practical configuration for the P3P problem (Gao et al., 2003). We can obtain up to 4 solutions for the pose using 3 points, and in the geometric approach, we choose the solution that results in smallest squared re-projection error for the 4th point (Gao et al., 2003),

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} \sum_{\forall i} \|\mathbf{p}_Q^{(i)} - \mathbf{KMP}^{(i)}\|, \quad i \in \{1, 2 \ldots, n\} \tag{A.8}$$

where $\mathbf{M}$ is the sought world-to-camera transformation matrix, $\mathbf{M}^*$ is its best estimate, $\mathbf{K}$ is the intrinsic matrix, $\mathbf{p}_Q^{(i)}$ is the ith feature point at the query image and $\mathbf{P}^{(i)}$ is its corresponding 3D coordinate.

In reality, the set of 2D-3D correspondences $\hat{C}$ can be corrupted by outliers, so it is common to use a robust estimator together with PnP solvers. RANSAC (Fischler and Bolles, 1981) estimator is a popular choice, and in our work we use a generalization of the RANSAC estimator, MLESAC (Torr and Zisserman, 2000). MLESAC adopts the same sampling strategy as RANSAC to generate putative solutions, but chooses the solutions by maximizing the likelihood rather than just the number of inliers.

Finally, the 6-DoF camera pose can be obtained by means of the decomposition of $\mathbf{M}^*$ via (A.3).

Appendix B. Direct photometric-based camera pose estimation

This appendix explains the details of the three stages of the *direct* photometric-based camera pose estimation, namely, generation of synthetic views, direct photometric matching and coarse-to-fine search.

B.1. Generation of synthetic views

The reference 3D point cloud P_R does not have any color or intensity information, but this information can be retrieved from the reference image as follows. Firstly, we project 3D point clouds $P_R = [\mathbf{P}_R^{(1)}, \mathbf{P}_R^{(2)}, \ldots, \mathbf{P}_R^{(m)}]$ onto the reference image plane using (A.2) and get a set of 2D projections, $p = [\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \ldots, \mathbf{p}^{(m)}]$. This process is the same as Fig. A.13. Secondly, we use cubic interpolation to compute the intensity values for each 2D projection and assign the intensity values to the 3D point cloud as:

$$I(\mathbf{P}_R^{(i)}) \leftarrow f(\mathbf{p}_R^{(i)}, I_R), \quad I_R \in \mathbb{R}^2 \tag{B.1}$$

where I_R is the reference image, $\mathbf{p}^{(i)}$ is the ith 2D projection, $I(\mathbf{P}_R^{(i)})$ is the intensity value of the 3D point $\mathbf{P}_R^{(i)}$, and f is the cubic interpolation function. As a result, we assign intensity (or color) information to the 3D point cloud P_R.

Synthetic views can now be rendered by projecting the colored 3D point cloud using a transformation matrix $\mathbf{M}$ using (A.2), and the intensities of the synthetic view I_S can be obtained as:

$$I_S(\mathbf{KMP}_R^{(i)}) \leftarrow I(\mathbf{P}_R^{(i)}), \tag{B.2}$$

where $I(\mathbf{P}_R^{(i)})$ is the intensity value of the ith 3D point $\mathbf{P}_R^{(i)}$, $\mathbf{K}$ is the intrinsic matrix, $\mathbf{M}$ is the world-to-synthetic-view transformation, and $I_S(\mathbf{KMP}_R^{(i)})$ is the intensity value of the projection of the 3D point $\mathbf{P}_R^{(i)}$ at the synthetic frame. Synthetic views are quickly rendered by the standard computer graphics procedure of surface splatting (Zwicker et al., 2001).

B.2. Direct photometric matching

The *direct* photometric-based approach (Tykkälä et al., 2013) is defined as a direct minimization of the cost function in the space of 6D camera pose, and in the cost function it compares the pixel intensities of the query image I_Q and rendered synthetic view I_S from the colored 3D point cloud (Tykkälä et al., 2013). The task is to find the best relative camera transform $\mathbf{M}^*$ that minimizes the photometric error between query image I_Q and synthetic image I_S:

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} \mathrm{RES}(I_Q, I_S), \tag{B.3}$$

where, the photometric error is represented by a residual (RES) defined as

$$\mathrm{RES}(I_Q, I_S) = \frac{1}{\mu} \sum_{(u,v)\in I_S} (I_Q(u,v) - I_S(u,v))^2 \tag{B.4}$$

In (B.4) I_Q is the query image, the synthetic view I_S is generated by (B.2), and μ is the number of pixels in I_S.

To improve the robustness of the matching process, we smooth the query image I_S by using a Gaussian filter and then we use the smoothed version of query image in the image matching process. Moreover, we use the M-estimator to improve the matching process, since the M-estimator can be used for managing outliers when the residual vector is of sufficient length for statistical purpose (Huber, 2011). The main idea is to generate small weights for residual elements that are classified as outliers by analyzing the distribution of residual values. Inliers always have small residual values whereas outliers may have any error value. In our work, a median filter is used to find the median value among the residuals, $\mathrm{RES}(I_Q, I_S)$, then we give zero weights to all the residual values that are greater than the median value, and give normalized weights to the remaining residuals.

With the M-estimator, we can rewrite the residual (B.4) as the average of the weighted sum-of-square difference:

$$\mathrm{RES}(I_Q, I_S) = \frac{1}{\lambda} \sum_{\forall(u,v)} (E(u,v))^2 w(u,v) \tag{B.5}$$

where we apply the weights to the residual vector and compute the average of the weighted sum-of-square difference, and λ is the number of nonzero weights. The squared difference $E(u,v)$ and weights $w(u,v)$ are defined in (B.6) and (B.7) as follows:

$$E(u,v) = (I_Q(u,v) - I_S(u,v))^2, (u,v) \in I_S \tag{B.6}$$

where I_Q is the query image, I_S is the synthetic image, and E is the difference between the two images.

$$w(u,v) = \begin{cases} 0, & \text{if } E(u,v) > \theta \\ 1, & \text{otherwise} \end{cases} \tag{B.7}$$

where θ is the median value of $E(u,v)$ and $(u,v) \in I_S$.

B.3. Coarse-to-fine grid search

We use a two-step coarse-to-fine grid search to solve for the matrix $\mathbf{M}^*$ in (B.3). The coarse-to-fine grid search concatenates a search with a coarse step for the local minimum with a subsequent search with a finer step at the location of the previous minimum location. Given a reference image, we use the camera pose of the reference image as the starting point for grid search. The coarse-to-fine search is firstly applied to the translation, and based on the previous minimum, we then apply

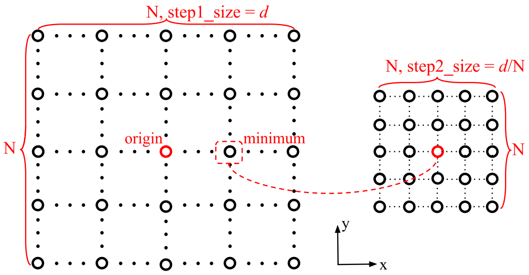


Fig. B.14. Coarse-to-fine grid search for translation. Grids are placed along x (toward to the right of the camera) and y (toward to the front of the camera) axis. Search the minimum within N steps of the step size d in a search grid, then apply a finer grid in the minimum point with another N steps of the size d/N.

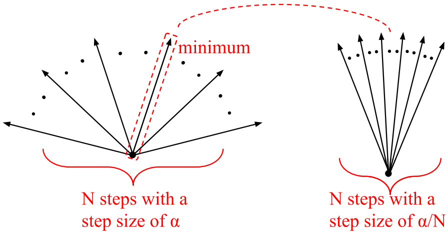


Fig. B.15. Coarse-to-fine grid search for orientation. For the selected axis (z axis, toward up of the camera), search by N steps of the angular size α, then refine the search by another N of the size α/N.

it to the orientation. The process of the coarse-to-fine grid search is illustrated in Figs. B.14 and B.15, and we describe the steps as follows:

Firstly, we take the orientation of the reference image for query image and start coarse-to-fine grid search for translation. There are 2 iterations in total. In the 1st iteration, we define a 2D grid along the x axis (towards the right of the camera) and y axis (toward the front of the camera) with a grid dimension of N and a step size of 10. A synthetic view I_S is generated for each grid point by (B.2), then we apply (B.5) to compute the residual value (RES) for this grid point. Then grid point with the minimum residual value is taken as the starting point for the 2nd iteration. In the 2nd iteration, we reduce the step size by 10 times and repeat the same procedure. In the end, we have estimated translation for query image. The above coarse-to-fine grid search for translation is illustrated in Fig. B.14.

Secondly, we fix translation of the query image and apply another coarse-to-fine grid search for orientation. We could search the optimal orientations along one or multiple axes. For our experiments, we search the optimal orientations along the z axis (toward up direction of the car), i.e. optimizing the yaw angle. The search procedure is similar to the one for translation. The process of the coarse-to-fine grid search for orientation is illustrated in Fig. B.15.

In our experiments, the both datasets consist of images captured by cameras mounted on cars and therefore there is mainly variation in the yaw angle for orientation. In our experimental setup, we choose to do orientation search only along the z axis. The full 6-DoF grid search would require a combination of the translation search (Fig. B.14) and three orientation searches (Fig. B.15).

In the process of generating a synthetic view I_S, a 3D point cloud is projected on a camera pose by (A.2). For each synthetic view in the grid search, we count the number of points projected inside the image frame. If the number of projected points is less than a threshold (100 in our experiments, see Table C.10), the synthetic view is considered as invalid. The invalid synthetic view is skipped in the grid search. If all synthetic views are invalid, the grid search fails to give a camera pose estimate.

Table C.8
Details of the test platform.

Processor	Intel i7CPU 2.70 GHz
OS	Ubuntu 16.04
Memory	32 GB
SW Env.	Matlab

Table C.9
Average time performance of the evaluated methods with a single query and a single reference image. Note these two original papers (Tykkälä et al., 2013; Pascoe et al., 2017) were designed for slam problem, but we modified the algorithms to adjust to our problem, and we implemented them in a laptop without utilizing GPU and multi-threads.

	KITTI	Oxford RobotCar
FB (Kim et al., 2014)	0.06 s	0.08 s
PM (Tykkälä et al., 2013)	1.23 s	4.82 s
MI (Pascoe et al., 2017)	1.34 s	5.15 s
HY (Combine FB and MI)	0.07 s	4.00 s

Appendix C. Implementation details and limitations

C.1. Platform and time performance

For reference purposes, we implemented and tested all the evaluated methods without utilizing GPU or multi-thread processing. The specifications of the platform and the programming language are shown in Table C.8. The average computing times are reported in Table C.9. In our implementation, the feature-based approach was the fastest one. The most time-consuming task for the photometric and mutual-information method was generation of synthetic views Appendix B.1. The computations are slower for the Oxford RobotCar dataset since point clouds are much larger. In addition, with the Oxford RobotCar dataset the feature-based method fails more frequently, and the HY method takes more mutual-information matching which is much slower.

C.2. Data preprocessing

With the KITTI Visual Odometry dataset (Geiger et al., 2012), we utilize the original 3D point clouds (LIDAR), ground truth pose data, and gray-scale images of each sequence. With the Oxford RobotCar dataset (Maddern et al., 2017), we also utilize the LIDAR scans, camera pose, and the left image from the trinocular stereo camera. However, the original 2D LIDAR data is saved as a single scan instead of an accumulated 3D point cloud as in the KITTI dataset. Therefore we applied two pre-processing steps to Oxford point clouds:

1. We converted the 2D LIDAR scans into a 3D point cloud by utilizing the toolkit provided by the authors.[2]
2. For efficiency, we reduced the number of images in each sequence by using every 10-th image and removed the start and final frames where the car was usually parked. The mean metric distance between two consecutive frames are shown in Table 2.

C.3. Parameters selection

The details of all parameters used in our experiments are shown in Table C.10.

[2] https://github.com/ori-drs/robotcar-dataset-sdk.

Table C.10
Method parameter values used in the experiments.

Feature-based (FB)	
Feature type	SURF
Photometric (PM)	
Min # of projected points	100
Mutual-information (MI)	
Min # of projected points	100
Grid search	
Translation	
Grid dimension	$d = 2 \times r$ meters, r is the uncertainty radius.
# of steps (1st iter)	10
Step length (1st iter)	$\frac{d}{10}$ meters
# of steps (2nd iter)	10
Step length (2nd iter)	$\frac{d}{100}$ meters
Orientation	
Search range	30 degrees
# of steps (1st iter)	10
Step size (1st iter)	3°
# of steps (2nd iter)	10
Step size (2nd iter)	0.3°

References

Alahi, A., Ortiz, R., Vandergheynst, P., 2012. Freak: Fast retina keypoint. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 510–517.

Ambai, M., Yoshida, Y., 2011. CARD: Compact and real-time descriptors. In: IEEE International Conference on Computer Vision. pp. 97–104.

Bay, H., Tuytelaars, T., Van Gool, L., 2006. Surf: Speeded up robust features. In: European Conference on Computer Vision. Springer, pp. 404–417.

Belongie, S., Malik, J., Puzicha, J., 2001. Shape context: A new descriptor for shape matching and object recognition. In: Advances in Neural Information Processing Systems. pp. 831–837.

Calonder, M., Lepetit, V., Strecha, C., Fua, P., 2010. Brief: Binary robust independent elementary features. In: European Conference on Computer Vision. Springer, pp. 778–792.

Castellanos, J.A., Tardos, J.D., 2012. Mobile robot localization and map building: A multisensor fusion approach. Springer Science & Business Media.

Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C., 2004. Visual categorization with bags of keypoints. In: Workshop on Statistical Learning in Computer Vision in European Conference on Computer Vision, Vol. 1. Prague, pp. 1–2.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition, Vol. 1. pp. 886–893.

Delmerico, J., Scaramuzza, D., 2018. A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots. Memory 10, 20.

Engel, J., Koltun, V., Cremers, D., 2018. Direct sparse odometry. IEEE Trans. Pattern Anal. Mach. Intell. 40 (3), 611–625.

Engel, J., Schöps, T., Cremers, D., 2014. LSD-SLAM: Large-scale direct monocular SLAM. In: European Conference on Computer Vision. Springer, pp. 834–849.

Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24 (6), 381–395.

Forster, C., Pizzoli, M., Scaramuzza, D., 2014. SVO: Fast semi-direct monocular visual odometry. In: IEEE International Conference on Robotics and Automation. pp. 15–22.

Friedman, J.H., Bentley, J.L., Finkel, R.A., 1977. An algorithm for finding best matches in logarithmic expected time. ACM Trans. Math. Software 3 (3), 209–226.

Froba, B., Ernst, A., 2004. Face detection with the modified census transform. In: IEEE International Conference on Automatic Face and Gesture Recognition. pp. 91–96.

Gao, X.-S., Hou, X.-R., Tang, J., Cheng, H.-F., 2003. Complete solution classification for the perspective-three-point problem. IEEE Trans. Pattern Anal. Mach. Intell. 25 (8), 930–943.

Geiger, A., Lenz, P., Urtasun, R., 2012. Are we ready for autonomous driving? the KITTI vision benchmark suite. In: IEEE Conference on Computer Vision and Pattern Recognition.

Geiger, A., Ziegler, J., Stiller, C., 2011. Stereoscan: Dense 3d reconstruction in real-time. In: Intelligent Vehicles Symposium. IEEE, pp. 963–968.

Gibbons, J.D., Chakraborti, S., 2011. Nonparametric statistical inference. In: International Encyclopedia of Statistical Science. Springer, pp. 977–979.

Guo, Z., Zhang, L., Zhang, D., 2010. Rotation invariant texture classification using LBP variance (LBPV) with global matching. Pattern Recognit. 43 (3), 706–719.

Huber, P.J., 2011. Robust statistics. Springer.

Irschara, A., Zach, C., Frahm, J.-M., Bischof, H., 2009. From structure-from-motion point clouds to fast location recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2599–2606.

Iscen, A., Tolias, G., Avrithis, Y., Furon, T., Chum, O., 2017. Efficient diffusion on region manifolds: Recovering small objects with compact CNN representations. In: IEEE Conference on Computer Vision and Pattern Recognition.

Kadir, T., Brady, M., 2001. Saliency, scale and image description. Int. J. Comput. Vis. 45 (2), 83–105.

Kim, H., Lee, D., Oh, T., Lee, S.W., Choe, Y., Myung, H., 2014. Feature-based 6-dof camera localization using prior point cloud and images. In: Robot Intelligence Technology and Applications 2. Springer, pp. 3–11.

Kitt, B., Geiger, A., Lategahn, H., 2010. Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme. In: Intelligent Vehicles Symposium. IEEE, pp. 486–492.

Klaser, A., Marszałek, M., Schmid, C., 2008. A spatio-temporal descriptor based on 3d-gradients. In: British Machine Vision Conference. British Machine Vision Association, 275–1.

Klein, G., Murray, D., 2007. Parallel tracking and mapping for small AR workspaces. In: IEEE and ACM International Symposium on Mixed and Augmented Reality. pp. 225–234.

Leutenegger, S., Chli, M., Siegwart, R.Y., 2011. BRISK: Binary robust invariant scalable keypoints. In: IEEE International Conference on Computer Vision. pp. 2548–2555.

Li, A.Q., Coskun, A., Doherty, S.M., Ghasemlou, S., Jagtap, A.S., Modasshir, M., Rahman, S., Singh, A., Xanthidis, M., OKane, J.M., et al., 2016. Experimental comparison of open source vision-based state estimation algorithms. In: International Symposium on Experimental Robotics. Springer, pp. 775–786.

Lienhart, R., Maydt, J., 2002. An extended set of haar-like features for rapid object detection. In: International Conference on Image Processing, Vol. 1.

Linegar, C., Churchill, W., Newman, P., 2016. Made to measure: Bespoke landmarks for 24-hour, all-weather localisation with a camera. In: IEEE International Conference on Robotics and Automation. pp. 787–794.

Lowe, D.G., 1999. Object recognition from local scale-invariant features. In: IEEE International Conference on Computer Vision. pp. 1150–1157.

Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 60 (2), 91–110.

Maddern, W., Pascoe, G., Linegar, C., Newman, P., 2017. 1 year, 1000km: The oxford robotCar dataset. Int. J. Robot. Res. 36 (1), 3–15.

Markley, F.L., Cheng, Y., Crassidis, J.L., Oshman, Y., 2007. Averaging quaternions. J. Guidance Control Dyn. 30 (4), 1193.

Matas, J., Chum, O., Urban, M., Pajdla, T., 2004. Robust wide-baseline stereo from maximally stable extremal regions. Image Vis. Comput. 22 (10), 761–767.

McDaid, A.F., Greene, D., Hurley, N., 2011. Normalized mutual information to evaluate overlapping community finding algorithms, arXiv preprint arXiv:1110.2515.

Mikolajczyk, K., Schmid, C., 2004. Scale & affine invariant interest point detectors. Int. J. Comput. Vis. 60 (1), 63–86.

Mishkin, D., Matas, J., Perdoch, M., 2015. Mods: Fast and robust method for two-view matching. Comput. Vis. Image Underst. 141, 81–93.

Miura, S., Hsu, L.-T., Chen, F., Kamijo, S., 2015. GPS Error correction with pseudorange evaluation using three-dimensional maps. IEEE Trans. Intell. Transp. Syst. 16 (6), 3104–3115.

Mori, G., Ren, X., Efros, A.A., Malik, J., 2004. Recovering human body configurations: Combining segmentation and recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, Vol. 2. II–II.

Muja, M., Lowe, D.G., 2009. Fast approximate nearest neighbors with automatic algorithm configuration.. Int. Conf. Comput. Vis. Theory Appl. 2 (331–340), 2.

Mur-Artal, R., Montiel, J.M.M., Tardos, J.D., 2015. ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE Trans. Robot. 31 (5), 1147–1163.

Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., Fitzgibbon, A., 2011a. Kinectfusion: Real-time dense surface mapping and tracking. In: IEEE International Aymposium on Mixed and Augmented Reality. pp. 127–136.

Newcombe, R.A., Lovegrove, S.J., Davison, A.J., 2011b. DTAM: Dense tracking and mapping in real-time. In: IEEE International Conference on Computer Vision. pp. 2320–2327.

NovAtel-Inc., 2019. span gnss inertial systems, Accessed: 2019-03-01, https://www.novatel.com/products/span-gnss-inertial-systems/.

Ohta, Y., Tamura, H., 2014. Mixed reality: merging real and virtual worlds. Springer Publishing Company, Incorporated.

Ojala, T., Pietikainen, M., Maenpaa, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. Pattern Anal. Mach. Intell. 24 (7), 971–987.

Oxford-Technical-Solutions-Ltd, OXTS-RT3000, Accessed: 2019-03-01, https://www.oxts.com/products/rt3000/.

Pascoe, G., Maddern, W., Tanner, M., Pinies, P., Newman, P., 2017. NID-SLAM: Robust monocular SLAM using normalised information distance. In: IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, HI.

Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A., 2007. Object retrieval with large vocabularies and fast spatial matching. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8.

Radenović, F., Tolias, G., Chum, O., 2016. CNN image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In: European Conference on Computer Vision.

Rosten, E., Drummond, T., 2006. Machine learning for high-speed corner detection. European Conference on Computer Vision 430–443.

Rublee, E., Rabaud, V., Konolige, K., Bradski, G., 2011. ORB: An efficient alternative to SIFT or SURF. In: IEEE International Conference on Computer Vision. pp. 2564–2571.

Sattler, T., Havlena, M., Schindler, K., Pollefeys, M., 2016. Large-scale location recognition and the geometric burstiness problem. In: IEEE Conference on Computer Vision and Pattern Recognition.

Scovanner, P., Ali, S., Shah, M., 2007. A 3-dimensional sift descriptor and its application to action recognition. In: ACM International Conference on Multimedia. pp. 357–360.

Song, Y., Chen, X., Wang, X., Zhang, Y., Li, J., 2016. 6-DOF image localization from massive geo-tagged reference images. IEEE Trans. Multimed. 18 (8), 1542–1554.

Strecha, C., Bronstein, A., Bronstein, M., Fua, P., 2012. Ldahash: Improved matching with smaller descriptors. IEEE Trans. Pattern Anal. Mach. Intell. 34 (1), 66–78.

Taylor, A.G., 2016. Develop microsoft hololens apps now. Springer.

Tola, E., Lepetit, V., Fua, P., 2010. Daisy: An efficient dense descriptor applied to wide-baseline stereo. IEEE Trans. Pattern Anal. Mach. Intell. 32 (5), 815–830.

Torr, P.H., Zisserman, A., 2000. MLESAC: A new robust estimator with application to estimating image geometry. Comput. Vis. Image Underst. 78 (1), 138–156.

Tuytelaars, T., Mikolajczyk, K., et al., 2008. Local invariant feature detectors: a survey. Found. Trends. Comput. Graph. Vision 3 (3), 177–280.

Tuytelaars, T., Van Gool, L.J., 2000. Wide baseline stereo matching based on local, affinely invariant regions.. In: British Machine Vision Conference, Vol. 412.

Tuytelaars, T., Van Gool, L., 2004. Matching widely separated views based on affine invariant regions. Int. J. Comput. Vis. 59 (1), 61–85.

Tykkälä, T., Comport, A.I., Kämäräinen, J.-K., 2013. Photorealistic 3D mapping of indoors by RGB-d scanning process. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 1050–1055.

Wen-Tsun, W., 1986. Basic principles of mechanical theorem proving in elementary geometries. J. Automated Reason. 2 (3), 221–252.

Zahn, C.T., Roskies, R.Z., 1972. Fourier descriptors for plane closed curves. IEEE Trans. Comput. 100 (3), 269–281.

Zhao, G., Pietikainen, M., 2007. Dynamic texture recognition using local binary patterns with an application to facial expressions. IEEE Trans. Pattern Anal. Mach. Intell. 29 (6).

Zwicker, M., Pfister, H., Van Baar, J., Gross, M., 2001. Surface splatting. In: Conference on Computer Graphics and Interactive Techniques. ACM, pp. 371–378.

PUBLICATION

II

A 3D map augmented photo gallery application on mobile device

J. Fu, L. Fan, K. Roimela, Y. You and V. Mattila

*2014 IEEE International Conference on Image Processing (ICIP)*2014, 2507–2511

DOI: 10.1109/ICIP.2014.7025507

A 3D MAP AUGMENTED PHOTO GALLERY APPLICATION ON MOBILE DEVICE

Junsheng Fu[⋆] [†] *Lixin Fan*[†] *Kimmo Roimela*[†] *Yu You*[†] *Ville-Veikko Mattila*[†]

[⋆] Tampere University of Technology, Department of Signal Processing, Finland
[†] Nokia Research Center, Media Technologies Lab, Finland
{ext-junsheng.1.fu, lixin.fan, kimmo.roimela, yu.you, ville-veikko.mattila}@nokia.com

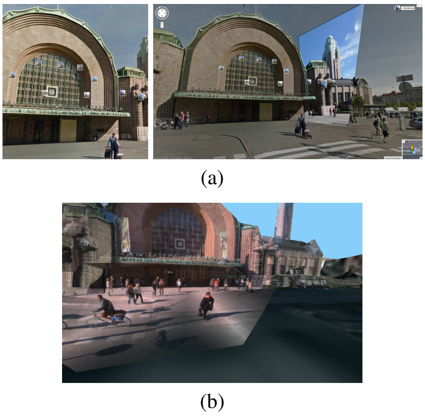
ABSTRACT

This paper proposes a 3D map augmented photo gallery mobile application that allows user to virtually transit from 2D image space to the 3D map space, to expand the field of view to surrounding environments that are not visible in the original image, and to change viewing angles among different global registered images during the image browsing. The processing of images consists of two main steps: in the first step, the client application uploads an image to the GeoImage Engine which extracts the geo-metadata and returns them back to the client; in the second step, the client application requests augmented content from server, and then renders 3D view of images on the screen of mobile devices.

***Index Terms*—** Mobile Image Applications, Augmented Reality, 3D Map

1. INTRODUCTION

Capturing an image with a camera phone and sharing the photo with a friend or on social media has gradually becoming part of our daily activities, because of the increasing penetration rate of mobile phones and popularity of image sharing service. Currently, map-based services also integrate location-based photo sharing functionality. For example shown in Fig. 1 (a), Google Map allows users to view floating thumbnails during street-view navigation, and once a thumbnail is selected, e.g. by mouse clicking, users can change the viewing angle from the street-view image to the 2D images. While it provides an interactive experience, this service is more gear to the augmentation of the street-view navigation, and thus, image browsing is somehow limited to 2D experience. In this paper, we propose a mobile application that provides users with a 3D map augmented image browsing experience by exploiting a back-end server to automatically compute global positions and orientations of images uploaded from the client application. With this mobile application installed, users can see from a mobile device screen where the images were exactly taken in the real word and view the image projection to the 3D building in the map model. One snapshot of the application is shown in Fig. 1 (b).



(a)

(b)

Fig. 1. (a) two screen shoots from Google map street-view service. Floating thumbnails in the street-view image indicates 2D images associated with the current scene. An enlarged image is overlaid on the street view, if the mouse pointer is hovering on a corresponding thumbnail. Note the overlay of thumbnail is based on 2D homograph transform and noticeable artifacts are often observed for non-planar scenes. (b) a screen shoot from our 3D map augmented photo gallery application. One user captured image is shown in a 3D map, and since the building facet and the ground are modelled separately, the mapped photo image gives more immersive 3D experience.

There are two challenging tasks in this kind of augmented reality applications. The first difficulty, for each uploaded photo, lies in the computation of the global 6 degree of freedom camera pose which consists of the position and orientation in a global coordinate system. The second demanding issue is related to the rendering of the user-captured images in the 3D map.

Related works are discussed as follows. With the progress of the structure from motion techniques, image browsing is not only limited in a 2D space and users can interactively move the viewing angle in the 3D space by seamlessly transitioning between different images. One application is Microsoft Photosynth [1], which can automatically computes each photo's viewpoint, generates a sparse 3D model of the scene, and calculates a smooth path through the camera pose for a set of given photo. With this path, Photosynth provides the experience of moving through a gliding motion and photos are sliced into multi-resolution pyramids for efficient access. While good scalability and impressive rendering effects have been demonstrated, the application relies on the feature matching among the user uploaded images, and calculates the camera poses in a local coordinate system instead of in a global coordinate system. To our best knowledge and surprise, there are only a handful of global camera pose based systems reported in the literature. In a more recent research by Zhang et al. [2] the approach is to match a user generated query image against a database of geo-tagged images with known global 6 degrees of freedom poses. Once a correct image match is made, the point to point correspondence between query and retrieved image is used to compute a homograph transformation which can be used to transfer pixel accurate tag information onto the query image. While good localization accuracy and efficiency are demonstrated, the overall system is more focused on localization applications instead of image browsing. As illustrated by Liu et al. [3], the mobile visual localization system can extract a comprehensive set of geo-context information from a single photo. While high localization accuracy and good scalability are demonstrated, the overall system is more gear to localization applications instead of image browsing. Our earlier work [4] utilizes the associate global camera pose to enrich the video playback experience.

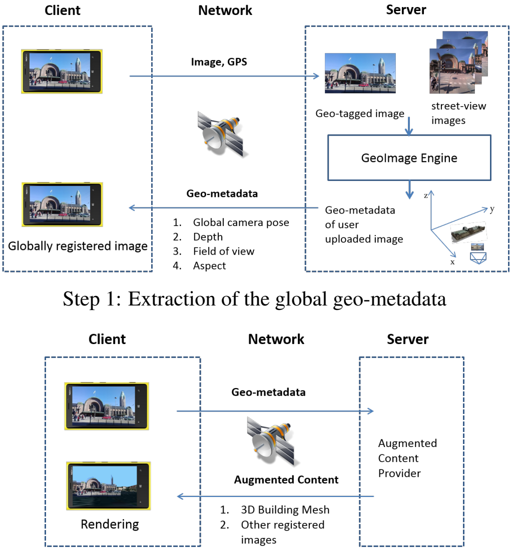
This paper illustrates a novel 3D map augmented photo gallery application that automatically uploads images from mobile clients to the server, extracts images' global camera poses and, consequently, enables augmented and interactive image browsing experiences with the augmentation of a 3D map.

2. SYSTEM FRAMEWORK

This section gives an overview of the system architecture and important functional modules, as shown in Fig. 2. The system framework consists of two main steps, named *extraction of global geo-metadata* and *client rendering*, and both steps are elaborated in this section.

2.1. Extraction of global geo-metadata

In the first step, the client uploads an image and its GPS to the server, the GeoImage Engine automatically extracts images geo-metadata and returns geo-metadata back to the client, see Step 1 in Fig. 2. The GeoImage Engine is an essential module, and it involves several important components, as shown in Fig. 3.



Step 1: Extraction of the global geo-metadata

Step 2: Client Rendering

Fig. 2. An overview of the system framework, which consists of two main steps.

- Once the image and its GPS are sent to the server, the GeoImage Engine starts to search and download closest 200 street-view images.
- Extract and compare the SIFT [5] features of both uploaded image and the street-view images, and then rank the street-view images according to the similarity to the uploaded one. Top k street-view images together with the uploaded image are used as input for 3D reconstruction.
- The 3D reconstruction module recovers the camera poses within a local coordinate system. This module uses Structure from Motion technique and we refer interested readers to [6, 7] for technique details.
- Based on the 3D reconstruction results, the depth range for the user captured image can be estimated. To find the camera pose in a global coordinate system, such as Earth-Centered Earth-Fixed (ECEF) system, we need to transform the local camera pose. Since registered street-view images have known camera poses in both local and global coordinate systems, a unique rigid transform is recovered [8] so that camera pose of user captured image can be mapped into the ECEF system.
- Finally, the GeoImage engine returns the client appli-

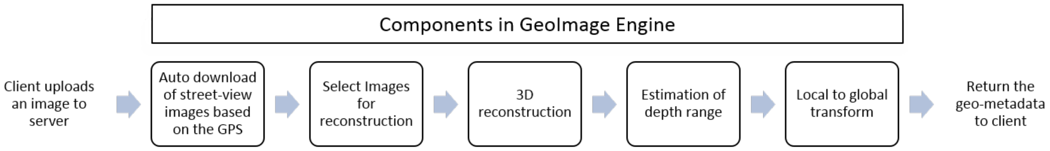

Components in GeoImage Engine
Client uploads an image to server
Auto download of street-view images based on the GPS
Select Images for reconstruction
3D reconstruction
Estimation of depth range
Local to global transform
Return the geo-metadata to client

Fig. 3. Data flowchart in GeoImage Engine.

cation with geo-metadata of the user uploaded image, including global camera pose, depth, field of view and aspect ratio.

2.2. Client Rendering

Once geo-metadata are returned from the GeoImage Engine, the client application requests nearby augmented content such as 3D map, building mesh, and other globally registered images from the content provider, see Step 2 in Fig. 2. Since the geo-metadata of the image and related augmented data are provided, it is possible to render everything within a unified global coordinate system.

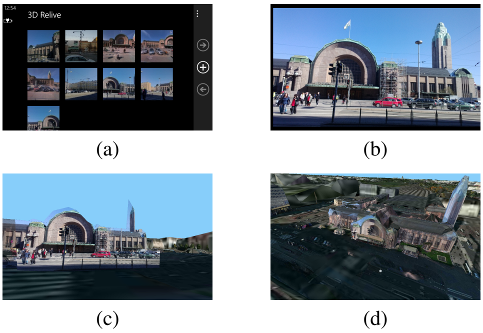
Our rendering algorithm for the images is based on view-dependent texturing [9]. To handle global geo-coordinates, all rendered content is first transformed to eye space, i.e., the local coordinate system of the current rendering camera. From the pose and projection parameters of each image, we calculate a per-image texture matrix that computes projected texture coordinates for image sampling. The texture matrix is passed to a pixel shader that also computes a per-pixel blending factor based on the angles between the original image ray and the current viewing ray, and the image ray and the normal vector of the surface being projected onto. The result RGBA pixels are then blended with the underlying map texture or other projected images.

Consequently, a number of intriguing image browsing experiences are now possible, and we demonstrate the mobile image application in a mobile phone as shown in Fig 4.

- Once the client application starts, users can see the photo gallery as the start screen, see Fig. 4 (a).

- Users can select a photo by clicking the image. Fig. 4 (b) shows one selected photo and the current view is in 2D image space.

- If users pinch in the image, a 3D map will be augmented in the client, and the viewing space would seamlessly transit from 2D image space to 3D map space, as shown in Fig. 4 (c). Moreover, users can see that the selected image is projected to the 3D building mesh based on its global camera pose.



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

Fig. 4. Here are four screen shoots from our 3D map augmented mobile image application. (a) start screen. (b) select one image from the photo gallery. (c) switch from the 2D image view to 3D map view when user pinches in the image, and the image is projected to the build mesh. (d) user can arbitrarily change the viewing angles. Currently, user is looking at the projected image from a bird view.

- Users can arbitrarily change the viewing angles, e.g. looking from right, left or even a bird view as shown in Fig.4 (d). If multiple images in the same area are tagged with global camera pose, users can select other images in 3D map space and seamlessly transits from current view to other images' views.

3. EXPERIMENTS

To evaluate the performance of the proposed system, we made two experiments. (1) Test the pipeline with the user captured mobile images in real case. (2) Test the pipeline by using the street-view images with the 6 degree of freedom ground truth camera pose.

In the first experiment, two users captured 147 images in Helsinki downtown area with Nokia Lumia 820 mobile phones. The client application automatically upload the image to the server for processing. A computer with the processor of Intel Core i7 CPU @3.4GHz and the memory of

16 GB, is used for back-end processing. On the server side, based on the image GPS recorded from the mobile device, the most closest 200 street-view images is used for feature matching. Secondly, Bundle Adjustment [7] is used to calculate the camera poses. Thirdly, other metadata, including depth, field of view and image aspect ratio are computed accordingly. Finally, all the metadata are transformed to ECEF coordinate system, and returned to the mobile client.

The experiments results for the 147 user captured images are shown in Table 1, in which about 35% of the user captured images are able to be successfully recovered. Failure modes are mainly due to the lack of reliable features in textureless regions e.g. skies or ground. We are exploring various techniques to still improve the performance of the proposed pipeline. The registration time is on average less than 5 minutes, and we found this registration time is acceptable for our designed use cases, in which the browsing of 3D augmented contents often does not immediately follow the photo taking action. This is especially true when users share a photo with friends through social media networks.

In order to evaluate the accuracy of the registered camera poses, we test the pipeline with Nokia Here street-view images that have ground truth camera pose. In the second experiment, 305 Nokia Here street-view images from Helsinki downtown area are used to test the pipeline. The global camera pose consists of camera's location and orientation, and the camera's orientation difference can be represented as follow:

$$P = |P\_rec - P\_ori|$$

where P_rec means the orientation of the recovered camera pose, P_ori means the orientation of the ground truth pose provided by Nokia Here map, and P indicates the difference between two orientations. In Fig. 5 (a), we use P to represent the orientation difference in degree, and $P \in [0, 180]$ degree. According to Fig. 5 (a), the recovered camera pose for these 305 street-view images have satisfactory accuracy of orientation, and the maximum orientation error among test images is less than 0.18 degree.

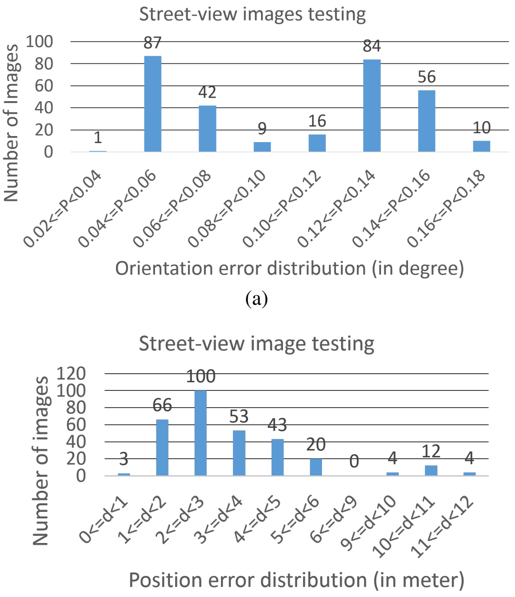
The location distance of the recovered and the ground truth pose can be calculated as follows:

$$d = |l\_rec - l\_ori|$$

in which l_rec means for location of the recovered camera pose in ECEF coordinate system, l_ori means the ground truth location of camera in ECEF coordinate system, and d is the distance between these two with the unit in meters. Fig. 5 (b) shows the histogram of position distances between the recovered camera pose and the ground truth pose. According to the Fig. 5 (b), around 93.4% of the street-view images have good accuracy and the errors are less than 6 meters. However, there are 5.2% images which has more than 10 meters errors due to wrong feature matches.

Table 1. Testing results for user captured images

Total Images	147
Registered images	52
Registration rate	35.37%
Max processing time (mins per image)	13.2
Min processing time (mins per image)	2.7
Average processing time (mins per image)	4.7



(a)

(b)

Fig. 5. Histogram of orientation errors and the position errors for 305 street-view images.

4. CONCLUSION AND FUTURE WORK

This paper presented a 3D map augmented photo gallery application on mobile devices, which shows that the ordinary image sharing experience can be greatly enriched by leveraging the associated geo-metadata. Compared to existing systems, this mobile application can seamlessly transit from 2D image space to 3D map space, expend the field of view of the image to surrounding environments that are not visible in the original one, and change of the viewing angles. The experiment results demonstrate satisfactory accuracy performance of the pipeline. In the future, we will explore more efficient recovery of camera poses, targeting on real-time application.

5. REFERENCES

[1] Noah Snavely, Steven M. Seitz, and Richard Szeliski, "Photo tourism: Exploring photo collections in 3d," in *SIGGRAPH Conference Proceedings*, New York, NY, USA, 2006, pp. 835–846, ACM Press.

[2] J. Zhang, A. Hallquist, E. Liang, and A. Zakhor, "Location-based image retrieval for urban environments," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*, Sept 2011, pp. 3677–3680.

[3] Heng Liu, Tao Mei, Jiebo Luo, Houqiang Li, and Shipeng Li, "Finding perfect rendezvous on the go: Accurate mobile visual localization and its applications to routing," in *Proceedings of the 20th ACM International Conference on Multimedia*, New York, NY, USA, 2012, MM '12, pp. 9–18, ACM.

[4] Junsheng Fu, Lixin Fan, Yu You, and Kimmo Roimela, "Augmented and interactive video playback based on global camera pose," in *ACM Multimedia*, 2013, pp. 461–462.

[5] David G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[6] Changchang Wu, "Towards linear-time incremental structure from motion," in *3DV-Conference, 2013 International Conference on*, June 2013, pp. 127–134.

[7] Changchang Wu, S. Agarwal, B. Curless, and S.M. Seitz, "Multicore bundle adjustment," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, June 2011, pp. 3057–3064.

[8] W. Kabsch, "A discussion of the solution for the best rotation to relate two sets of vectors," *Acta Crystallographica Section A*, vol. 34, no. 5, pp. 827–828, Sep 1978.

[9] Paul E. Debevec, Yizhou Yu, and George Borshukov, "Efficient View-Dependent Image-Based Rendering with Projective Texture-Mapping," in *Eurographics Symposium on Rendering/Eurographics Workshop on Rendering Techniques*, 1998, pp. 105–116.

PUBLICATION

III

Augmented and Interactive Video Playback Based on Global Camera Pose

J. Fu, L. Fan, Y. You and K. Roimela

*Proceedings of the 21st ACM International Conference on Multimedia*2013, 461–462
DOI: 10.1145/2502081.2502269

Augmented and Interactive Video Playback Based On Global Camera Pose

Junsheng Fu
Media Technologies Lab,
Nokia Research Center
Tampere, Finland
junsheng.fu@nokia.com

Lixin Fan
Media Technologies Lab,
Nokia Research Center
Tampere, Finland
lixin.fan@nokia.com

Yu You
Media Technologies Lab,
Nokia Research Center
Tampere, Finland
yu.you@nokia.com

ABSTRACT

This paper proposes a video playback system that allows user to expend the field of view to surrounding environments that are not visible in the original video frame, arbitrarily change the viewing angles, and see the superimposed point-of-interest (POIs) data in an augmented reality manner during the video playback. The processing consists of two main steps: in the first step, client uploads a video to the GeoVideo Engine, and then the GeoVideo Engine extracts the geo-metadata and returns them back to the client; in the second step, client requests POIs from server, and then the client renders the video with POIs.

Categories and Subject Descriptors

H.5.1 [**Multimedia Information Systems**]: Video

General Terms

Algorithms, Performance, Design, Human Factors

Keywords

Video Playback, Augmented Reality, Camera Pose

1. INTRODUCTION

Capturing a video clip with a camera phone and sharing it with a friend later on has gradually becoming part of our everyday activities, because of the increasing penetration rate of mobile phones and popularity of video sharing services such as “YouTube”. The demo presented in this paper shows that such an ordinary *video capturing* and *sharing* experience can be greatly enriched by leveraging geo-metadata associated with every video frame (see supplementary video for an example). In particular, we demonstrate that by exploiting associated *global camera pose* information,

- video playback can be enhanced with augmented reality contents;
- video playback becomes an interactive and intriguing experience.

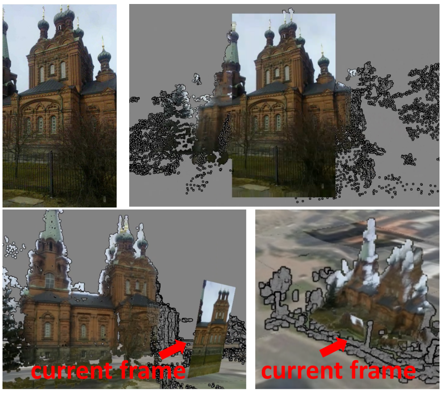


Figure 1: Three different kinds of video playback modes are presented. Ordinary mode (*upper left*): video is played back as same as the captured video; Expanded view mode (*upper right*): user can expend the filed of view to surrounding environments that are not visible in the original video frame; Map view mode (*bottom left and right*): camera motion of the video is highlighted in the map and viewing angles can be arbitrarily changed by the user.

To our best knowledge and surprise, this kind of geo-location enhanced video playback experience has not been fully pursued and there are only a handful of related systems reported in the literature. For instance, Vidmap [1] is a video uploading and sharing service which associates video frames with a 2D GPS location. When playing back video frames, corresponding GPS locations are synchronously displayed on a map. The service provides GPS locations in 2D only and makes no attempt to recover 6 degree-of-freedom (6-DoF) camera poses. In a more recent research by Liu et al. [2], the mobile visual localization system extracts a comprehensive set of geo-context information from a single photo. While high localization accuracy and good scalability are demonstrated,the overall system is more gear to localization applications instead of video playing back.

This paper illustrates a novel video processing system that automatically extracts global camera pose from uploaded video (Step 1 in Figure 2) and, consequently, enables augmented and interactive video playback experiences (Step 2 in Figure 2).

ACM MM 13 October 21-25, 2013, Barcelona, Spain.

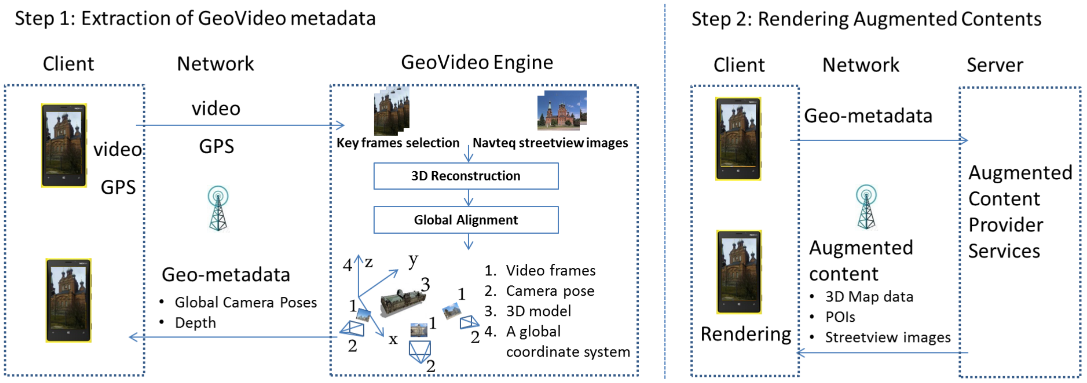


Figure 2: Overview of the proposed Augmented and Interactive Video Playback System.

2. SYSTEM FRAMEWORK

This section gives an overview of the system architecture and important functional modules.

2.1 Extraction of GeoVideo metadata

After client uploads a video to the GeoVideo Engine, the GeoVideo Engines automatically extract the Geo-matadata associasted with the uploaded video. The GeoVideo Engine involves two important components, namely, *3D Reconstruction* and *Global Alignment* (see Figure 2 *Step 1*). The *3D Reconstruction* module reconstructs 3D point clouds from selected key video frames and simultaneously recovers camera poses within a local coordinate system. This module uses Structure from Motion technique and we refer interested readers to [3, 4, 5] for technique details.

The *Global Alignment* module, aims to find global camera pose of key video frames by aligning nearby Navteq street view images with reconstructed 3D scenes. Since registered Navteq images have known camera poses in both the local and global coordinate systems, an unique rigid transform is recovered so that camera poses of key frames' can also be mapped into the global coordinate system. Camera poses of the rest of video frames can be estimated by adopting Efficient Perspective-n-Point Camera Pose Estimation method [6].

2.2 Rendering Augmented Contents

Once geo-metadata are returned from the GeoVideo engine, the client application may request nearby augmented content such as 3D map data, point-of-interest (POI) data, street-view panorama images etc from corresponding network services. Since all camera poses of original video frames and augmented data are provided, it is straightforward to render everything within a unified global coordinate system. Consequently, a number of intriguing video playing back experiences are now possible:

- *Expand Your View* (see Figure 1 *upper right*): During the playback of a GeoVideo, the field of view (FOV) of every video frame can be extended to 360 degree by using nearby panorama images. This rendering is possible because panorama images or 3D city models are often tagged with GPS information.

- *Arbitrary Change of Viewing Angles* (see Figure 1 *bottom left and right*): Users can highlight the camera motion in a map mode and the client application allows arbitrary change of viewing angles. Furthermore,the system allows users to easily switch between *ordinary view mode* and *map view mode*.

- *POIs-augmented Video Playback*: Based on point-of-interest (POIs) and associated geo-metadata, it is possible to augment each video frame with nearby POIs data. During the playback of a video, the change of camera poses gives rise to corresponding change in the rendered POI data, thus creating augmented-reality experience.

3. CONCLUSION

The demo presented in this paper shows that an ordinary *video capturing* and *sharing* experience can be greatly enriched by leveraging geo-metadata associated with every video frame. Compared to existing video playback systems, the presented system allows users to expend the filed of view to surrounding environments that are not visible in the original video. The system also allows arbitrarily change of the viewing angle. What is more, point-of-interest data can be superimposed on original video frames in an augmented reality manner and camera motion can be viewed in a map mode.

4. REFERENCES

[1] "Vidmap: http://www.vidmap.de/."

[2] H. Liu, T. Mei, J. Luo, H. Li, and S. Li, "Finding perfect rendezvous on the go: accurate mobile visual localization and its applications to routing," in *Proceedings of the 20th ACM international conference on Multimedia*, MM '12, (New York, NY, USA), pp. 9–18, ACM, 2012.

[3] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3d," *ACM Trans. Graph.*, vol. 25, pp. 835–846, July 2006.

[4] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 32, no. 8, pp. 1362–1376, 2010.

[5] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski, "Towards internet-scale multi-view stereo," in *CVPR*, 2010.

[6] F. Moreno-Noguer, V. Lepetit, and P. Fua, "Accurate non-iterative o(n) solution to the pnp problem," 2007.

PUBLICATION

IV

Indoor Objects and Outdoor Urban Scenes Recognition by 3D Visual Primitives

J. Fu, J.-K. Kämäräinen, A. G. Buch and N. Krüger

Computer Vision - ACCV 2014 Workshops. Ed. by C. Jawahar and S. Shan. 2015, 270–285
DOI: https://doi.org/10.1007/978-3-319-16628-5_20

Indoor Objects and Outdoor Urban Scenes Recognition by 3D Visual Primitives

Junsheng Fu[13] Joni-Kristian Kämäräinen[1] Anders Glent Buch[2]
Norbert Krüger[2]

[1] Vision Group, Tampere University of Technology, Finland, http://vision.cs.tut.fi
[2] CARO Group, University of Southern Denmark, Denmark, http://caro.sdu.dk
[3] Nokia Research Center, Finland

Abstract. Object detection, recognition and pose estimation in 3D images have gained momentum due to availability of 3D sensors (RGB-D) and increase of large scale 3D data, such as city maps. The most popular approach is to extract and match 3D shape descriptors that encode local scene structure, but omits visual appearance. Visual appearance can be problematic due to imaging distortions, but the assumption that local shape structures are sufficient to recognise objects and scenes is largely invalid in practise since objects may have similar shape, but different texture (e.g., grocery packages). In this work, we propose an alternative appearance-driven approach which first extracts 2D primitives justified by Marr's primal sketch, which are "accumulated" over multiple views and the most stable ones are "promoted" to 3D visual primitives. The 3D promoted primitives represent both structure and appearance. For recognition, we propose a fast and effective correspondence matching using random sampling. For quantitative evaluation we construct a semi-synthetic benchmark dataset using a public 3D model dataset of 119 kitchen objects and another benchmark of challenging street-view images from 4 different cities. In the experiments, our method utilises only a stereo view for training. As the result, with the kitchen objects dataset our method achieved almost perfect recognition rate for ±10° camera view point change and nearly 90% for ±20°, and for the street-view benchmarks it achieved 75% accuracy for 160 street-view images pairs, 80% for 96 street-view images pairs, and 92% for 48 street-view image pairs.

1 Introduction

Over the past few decades, object and scene recognition have achieved great success using 2D image processing methods. Recently, with the increasing popularity of Kinect sensors and the emergence of dual-camera mobile phone, researchers are motivated to approach the traditional image recognition problem with 3D computer vision methods. Compared with the successful 2D methods, 3D approaches are not limited to image 2D appearance as the cue for detection and recognition [1, 2]. A number of 3D methods for object and scene recognition have been proposed [3–5] to extract global or local shape descriptors that

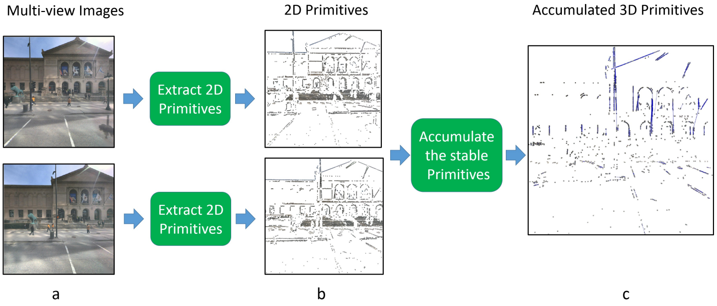


Fig. 1: Construct the 3D primitives from Multi-view images.

encode scene structure, however, they do not take the advantage of 2D visual appearance, e.g. colour and texture.

In accord with the recent trend of 3D object detection and recognition research, we propose in this paper an approach that utilizes both the 2D appearance and 3D structure from the multi-view images. The most important and novel processing of the proposed method, in our view, is the construction of the 3D primitive, i.e. 3D classified features derived from multi-view images. Fig. 1 shows the work-flow of the 3D primitive construction: Firstly, for each multi-view input, the pipeline computes the 2D visual primitives [6] using the intrinsic dimension by Kalkan et al. [7]. Secondly, the stable 2D primitives are matched across multi-view images and triangulated to 3D primitives, as shown in Fig. 1 c (see Section 3 for details). Then the 3D primitives are used for matching 3D objects primitives stored in a database.

To evaluate the proposed method, we tested our pipeline with both indoor objects and outdoor urban scenes. With the indoor objects dataset, our method achieved almost perfect recognition rate for ±10° camera view point change and nearly 90% for ±20°, and for the real world street-view dataset from 4 different cities, our method achieved 75% accuracy for 160 street-view images pairs, 80% for 96 street-view images pairs, and 92 % for 48 street-view image pairs.

Our main contributions are as follows:

- A novel 3D primitive extraction method for object recognition: 2D appearance primitives are extracted and promoted to 3D based on matching results across multi-view images.
- A simple random sampling based recognition to match observed 3D primitives to database objects. The training is based on a single recorded view.
- Novel results on the effect of primitive accumulation vs. no accumulation and 3D matching vs. 2D matching for object recognition in 3D.

- A semi-synthetic benchmark dataset and toolkit of 3D graspable kitchen items captured in the KIT.[1]. This can be used for further analysis in a controlled environment, and the code for rendering novel KIT object views will be made publicly available.
- A real benchmark dataset of stereo street views, which can be used for performances analysis in real conditions.

This paper is structured as follow. Firstly, the related work is presented in Section 2. Then, Section 3 and Section 4 explain the process of constructing 3D primitives from 2D primitives and the matching process of the 3D primitives. Section 5 illustrates the experiment results from both indoor objects database and outdoor street-view images from 4 different cities. Finally, we conclude in Section 6.

2 Related work

The object detection and recognition approaches can be roughly divided into 2D-to-2D (genuine 2D), 3D-to-2D (or 2D-to-3D) and 3D-to-3D (genuine 3D) methods, where the first term defines whether a model (and training data) are 2D or 3D and the latter whether objects are detected from 2D or 3D images. The most successful approach is part-based: local features are extracted and the object described as the parts and their location. Successful results have been reported for detection of visual classes and specific objects in 2D-to-2D [1, 2] and 3D-to-2D [8–10], and many of the methods provide state-of-the-art classification accuracy on common benchmarks.

Our main interest, however, are genuine 3D methods which have not yet reached a mature stage as the aforementioned methods. Next, we give a brief survey on the most recent works, but omit methods based on global description (e.g., [11]), those using temporal information [12, 13] and those tailored for a specific application, such as 3D face recognition [14, 15].

Two notable works related to our method are the ones by Papazov and Burschka [16] and Drost et al. [17]. Papazov and Burschka utilise a random sample principle while Drost et al. use Hough-like voting, but the main commonality is in the fact that they both directly use 3D point clouds, which ties their methods to the selected 3D capturing method. We use local primitives extracted from 2D RGB images. Similar vision primitives were used in Detry et al. [18] ([19]), but their method do not retain 3D structure, and recognition is performed by Markov process message passing utilising pairs of the primitives similar to [17].

The popular 2D interest point detectors and descriptors have also been extended to 3D, for example 3D SURF by Knopp et al. [20], local surface histograms [21] in Pham et al. [22], HOG and DoG by Zaharescu et al. [23] and kernel descriptors [24]. Special 3D shape detectors and descriptors have also been proposed [25, 26] along with neighbourhood processing to improve the robustness of shape descriptors [3, 5]. There are many local 3D shape descriptors (see [27,

[1] http://i61p109.ira.uka.de/ObjectModelsWebUI/

28]), but their main limitation is that they select the points based on local shape information and discard appearance which, after all, is the low-level source of information in the human visual system and used in the Marr's primal sketch [6]. The shape descriptors have been recently evaluated in [4]. One exception is Lee et al. [29] who utilise lines, but that is particularly suitable for their objects of interest (boxes). Hybrids of 3D shape and 2D texture descriptors were proposed by Hu and Zhu [30] and Kang et al. [31].

3 Constructing 3D primitives from 2D primitives

The visual primitives used in this work derive from the primitives found in various layers of the "deep vision hierarchy" [32]. Starting from the pixels (retinal image) we extract low level primitives which are re-sampled (added), deleted, combined (grouped) and promoted through bottom-up processing in the hierarchy. We refer to the operations with a single term, "accumulation". Various computational models of the hierarchy have been proposed [33–35]. out of which we adopt the "cognitive vision model" hierarchy by Pugeault et al. [35]. The main goal of their hierarchy is a symbolic 3D description of a scene, but we form primitives that construct a part-based 3D object model.

On the lowest hierarchy level, 2D primitives are extracted from the left and right images of a stereo pair (see Fig. 1). The primitives are extracted on a regular spatial grid where circular patches are extracted and assigned to one of four low-level classes: a constant colour region, edge/line, junction or texture. The classification is based on computational intrinsic dimensionality [7]. The computational intrinsic dimension, ifD, defined by a real number f measures the effective texture patch dimension similar to the fractal dimension [36], but can be computed fast with linear quadrature filters [37]. The ifD space forms a triangular region where basic perceptual classes map to distinct locations (Fig. 2):

- Constant colour: $ifD \approx i0D$
- Edge/line: $ifD \approx i1D$
- Junction: $i1D << ifD < i2D$
- Textured region $ifD \approx i2D$

The extracted 2D primitives are encoded as

$$\boldsymbol{\pi} = (\boldsymbol{x}, \theta, \phi, \boldsymbol{c}) \tag{1}$$

where $\boldsymbol{x}$ is the 2D image position, θ is the local orientation angle of an edge or line, ϕ is the local phase of an edge/line, and $\boldsymbol{c}$ is the RGB colour vector of the left, middle and right edge colours.

The accumulation of 2D primitives to 3D primitives $\boldsymbol{\Pi}$ is based on multiple views with known calibration: $accumulation : (\boldsymbol{\pi}, \boldsymbol{\pi}') \rightarrow \boldsymbol{\Pi}$. In order to be promoted, the 2D primitive descriptors—colour, orientation and phase—must match, the primitives must lie on their corresponding epipolar lines, and finally the spatial constraints must hold. For putative matches for a primitive $\boldsymbol{\pi}$ at $\boldsymbol{x}$ in the left image, the epipolar line $\boldsymbol{x}' \in l' = \boldsymbol{e}' \times \boldsymbol{H}_\pi \boldsymbol{x}$, where $\boldsymbol{e}' \times \boldsymbol{H}_\pi = \boldsymbol{F}$ is the fundamental matrix [38], in the right image is searched for $\boldsymbol{\pi}'$. Since the

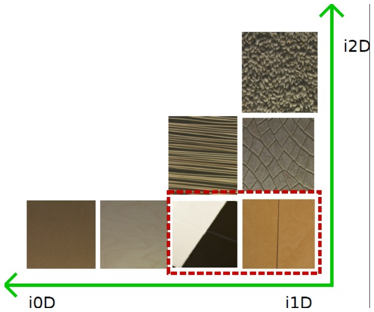


Fig. 2: Texture characterisation in the intrinsic dimension space [7], the 2D line and edge primitives used in our work marked with the dashed line.

2D primitives are computed sparsely on a grid, the matches within the distance of 1.5 times the patch size are accepted. The accumulated 3D primitives are encoded as

$$\boldsymbol{\Pi} = (\boldsymbol{X}, \boldsymbol{n}, \Theta, \Phi, \boldsymbol{C}) \tag{2}$$

where $\boldsymbol{X}$ is the 3D location in space, $\boldsymbol{n}$ is the surface normal, Θ the line/edge orientation, Φ the line/edge phase and $\boldsymbol{C}$ the colour vector constructed by the weighted average of the corresponding 2D colours.

In this work, we use the line/edge primitives (see Fig. 2). The 2D primitive extraction can be adjusted by three quadrature filter parameters [37]. The first parameter is the highest filter frequency (or image resolution). The second parameter is the minimum required energy within the circular patches (normalised to $[0,1]$) and the third parameter is the maximum variance (normalised to $[0,1]$), i.e. whether primitives must come from clearly isolated points (low variance). The descriptor match is a weighted sum of colour (weight 0.5), orientation (0.3) and phase (0.06) differences, all normalised to $[0,1]$, and the match threshold set to 0.3. Moreover, a spatial constraint, "external confidence", similar to stereo algorithms was added to ensure that the accepted 3D primitives are supported by their neighbourhood. By changing the values of the parameters we can affect the number of extracted 2D and 3D primitives and their robustness. Several settings are demonstrated in Table 1 for the first 12 KIT objects.

For the setting 1 approximately 50% of the 2D primitives are promoted. For other settings, the number of 2D primitives is much larger, but due to the accumulation there is not much difference between the number of 3D primitives for the settings 1-3. This is further illustrated in Fig. 3 where the 3D primitives (bottom) look alike for all settings. Note, however, that for Setting 2 and Setting 3 the new primitives are less reliable and therefore more noise appears. By using higher frequencies (a larger image), the number of primitives increases "naturally", i.e., more details are added to places where also the depth informa-

Table 1: Various 3D primitive extraction Parameter settings and the corresponding numbers of produced 3D primitives.

Parameter	*Setting 1*	*Setting 2*	*Setting 3*	*Setting 4*
Image size	300x300	300x300	300x300	400x400
Min. energy	0.4	0.4	0.4	0.4
Max. variance	0.2	0.6	0.2	0.2
Ext. conf.	0.1	0.1	-1.0	0.1

Object	*Setting 1 (2D)*	*Setting 1*	*Setting 2*	*Setting 3*	*Setting 4*
OrangeMarmelade	324	120	243	219	244
BlueSaltCube	410	251	326	315	433
YellowSaltCube	380	201	289	293	338
FruitTea	282	168	258	227	265
GreenSaltCylinder	246	72	158	166	140
MashedPotatoes	424	223	374	329	387
YellowSaltCylinder	355	168	236	247	329
Rusk	503	234	393	303	381
Knaeckebrot	372	186	269	242	300
Amicelli	414	276	384	384	509
HotPot	376	131	200	216	193
YellowSaltCube2	380	210	278	303	396
Avg.	372	187	284	270	326

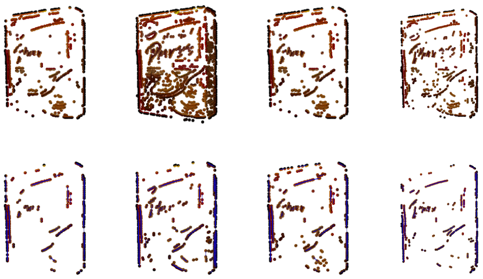


Fig. 3: Top: extracted 2D primitives (stereo left) with Settings 1-4 from the left to right. Bottom: the corresponding 3D primitives after the accumulation.

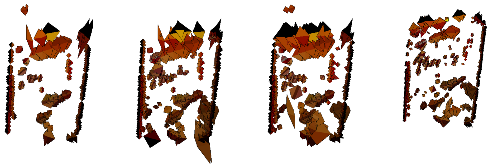


Fig. 4: The 3D primitives at the bottom of Fig. 3 re-drawn using the detected scales. See the last paragraph of Section 3 for details.

tion is reliable. That is illustrated Fig. 4, where 3D primitives are plotted in 3D space with their detected scale.

4 Matching 3D primitives

The 3D primitive based object description in Section 3 represents object appearance in the primitive descriptors Θ, Φ and $\boldsymbol{C}$ and object location in the 3-vectors $\boldsymbol{X}$. The two popular approaches to match descriptors in space are voting and random sampling. A variant of the random sampling appears in Papazov and Burschka [16] and voting (Hough transform) in Drost et al. [17].

The random sampling and voting have certain distinct properties as compared to each other. In the voting approach every primitive is processed once and they cast votes for multiple objects and for multiple poses. The best hypothesis is the one with the highest number of votes. A disadvantage is the size of the vote (accumulator) space, which can become huge without coarse discretisation. In the sampling approach, no accumulation is needed since every random sample generates one hypothesis of an object and its pose. The obvious disadvantage is that the required number of random samples may be large. In other words, the voting is more storage intensive and the sampling more computationally intensive. There exists studies to improve storage requirements and to reduce the number of samples (e.g., [39]), but in this work we select the sampling approach due to its simplicity.

4.1 Random sampling based matching

We randomly sample from the primitives of an object model i (object database), select corresponding primitives from an observed scene, and then compute the transformation $\boldsymbol{T}$ which brings the observed scene and database model primitives in correspondence. The method is similar to Papazov and Burschka [16], except that they directly use dense point cloud points which are sensitive to a selected 3D acquisition process. Additionally, to avoid computational explosion

Algorithm 1 Random sample consensus matching.

```
Compute the match matrix between each observed primitive $\boldsymbol{\Pi}_{i=1...N}$ and each model primitive $\boldsymbol{\Pi}_{i=1...M}$: $\boldsymbol{D}_{N \times M}$.
Sort and select the K best matches for each observation primitive $\rightarrow \hat{\boldsymbol{D}}_{N \times K}$.
for $R$ iterations do
    Randomly select 3 observation primitives from $1 \ldots N$ and their correspondences in $1 \ldots K$ in $\hat{\boldsymbol{D}}_{N \times K}$.
    Estimate the linear 3D transformation (isometry/similarity) $\boldsymbol{T}$ using the Umeyama method [40].
    Transform the all $N$ observation primitives to the model space with $\boldsymbol{T}$.
    Select the geometrically closest matches (within the $K$ best) and compute the match score $s$.
    Update the best match ($s_{best}$, $\boldsymbol{T}_{best}$) if necessary.
end for
Return $s_{best}$ and $\boldsymbol{T}_{best}$.
```

(every observation point is a candidate match to every model point), they utilise heuristics. Our method selects the best match using the 3D primitive descriptors. To estimate the 3D transformation (isometry) we use the linear method by Umeyama [40]. A high level algorithm for our matching method is given in Algorithm 1.

There are two important considerations for Algorithm 1: the number of iterations R and a method to compute the match score s. Since the colour plays the most important role in the accumulation, we omit Θ and Φ and use the colour vector $\boldsymbol{C}$ to compute the match matrix $\boldsymbol{D}$. $\boldsymbol{C}$ is a 9-vector of the RGB values for the edge/line left, middle and right which are uniquely defined. The match is the Euclidean distance between the vectors which is fast to compute. Also the colour covariances are available, but using them is computationally inefficient. L^2-normalisation makes the colour descriptors semi illumination invariant.

The number of iterations R is an important parameter since a sufficient number of samples is needed to guarantee that the correct combination is found with high confidence. To derive a formula for R we can consider the ideal case that each N observation point has a correct match in the model. The total number of points is not important, but the number of possible candidates. In Algorithm 1 this is K and we further assume that a correct correspondence is within the K best matches. Now, the probability of randomly selecting a correct combination of three point correspondences (the minimum for 3D isometry/similarity estimation) is

$$P(K) = \frac{1}{K} \cdot \frac{1}{K} \cdot \frac{1}{K} \quad . \tag{3}$$

Note that this would be $1/K(K-1)(K-2)$ if the points are shared. The probability that after R iterations no correct triplets have been drawn is $(1-P(K))^R$, and thus, the probability that at least one correct has been drawn is $1-(1-P(K))^R$. The analytical formula for the number of samples in order to pick at least one correct match with the probability P_S is

$$R = \frac{\log(1 - P_S)}{\log(1 - P(K))} \ . \tag{4}$$

For example, with $P_S = 0.9$ (90% confidence level), we get $R = 287$ for $K = 5$ and $R = 2302$ for $K = 10$. In practise, some primitives have no matches at all, but on the other hand, representation is typically dense in the most informative areas and any primitive near the correct one may succeed. In any case, K should not be more than 10 to limit computational burden ($R \leq 2000$).

To select the best strategy to compute the match score s, we run preliminary tests with the first 12 objects in the KIT dataset (see Table 2 for the results). More details are in Section 5, but here we focus only on the recognition accuracy. The rank order statistics rules, such as *median matching*, are superior due to their robustness to outliers and still computationally affordable. There is no major differences between the median (best 50%) and best 25%, with the number of samples doubled (2× iterations) and isometry vs. similarity, and therefore we selected the median rule. Note that the reverse matching (from models to the scene), is clearly inferior.

Table 2: Recognition accuracies for the first 12 KIT objects using variants of the match score s in Algorithm 1. $K = 10$ best matches and $R = 1000$ random samples (Setting 1, pure chance 8%).

s Method	*El-Az* 5°	*El-Az* 10°	*El-Az* 20°	*El-Az* 30°	*El-Az* 40°
Mean match	84%	74%	50%	34%	18%
Med match	100%	100%	98%	77%	46%
Med match (reverse)	100%	97%	65%	49%	28%
Best25% match	100%	100%	93%	70%	44%
Med match (2× iters)	100%	100%	98%	78%	46%
Med match (simil.)	100%	100%	96%	77%	45%

5 Experiments

In this Section, we evaluate our pipeline with both the **indoor objects dataset** and the **outdoor urban street-view images**.

A dataset was collected in Karlsruhe Institute of Technology (KIT): KIT Object Models Web Database[2]. The KIT dataset provides full high-quality 3D models, so we use the KIT dataset as the indoor objects database for testing the pipeline. For evaluation, we implemented a synthetic view generator that

[2] http://i61p109.ira.uka.de/ObjectModelsWebUI/

can be used to evaluate methods in controlled view points and illumination. To further evaluate the robustness of our pipeline, we gathered 160 street-view images pairs with the known camera poses from 4 different cities. The datasets and experiment results are discussed in the following two Subsections.

5.1 Indoor object dataset

Toolkit for semi-synthetic KIT Objects – The KIT object dataset contains 119 3D captured kitchen items (marmalade packages, mugs, tea packages etc.) suitable for robot grasping and manipulation [41] and stored as high-quality textured 3D polygon models. Using the KIT models (Fig. 5) we provide a public toolkit to generate arbitrary views points, ground truth, and benchmark recognition algorithms.

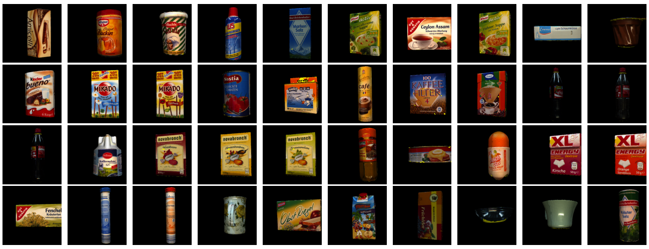


Fig. 5: Examples of the 119 KIT object models in frontal (training) pose. Note that some objects differ only by details in their appearance (colour or texture).

The toolkit was used to render the training images in roughly frontal pose (Fig. 5), automatically adjust the camera distance to fit objects' bounding boxes to the visible image area, generate stereo pairs (Fig. 6) and output the stereo camera matrices and bounding box world coordinates.

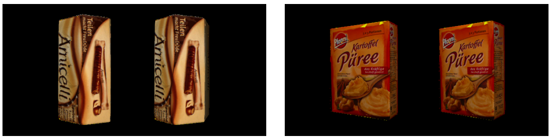


Fig. 6: The stereo pair frontal views of "Amicelli" (left) and "MashedPotatoes" (right). The camera baseline is fixed to 50 world units (1wu ≈ 1mm).

For our experiments, the object database (training set) was made by storing primitives from only one view per object: the frontal views shown in Fig. 5. The test set images were generated by geometrically transforming the same objects by adjusting the camera azimuth and elevation angles. A total of five different test sets were generated using gradually increasing angles: $\{-40°, -30°, \ldots + 40°\}$ This results to 9 test images per object and $119 \times 9 = 1071$ images in total for each test set. The test sets are referred to as Ez-Al-5° ... Ez-Al-40°. The two extremal test set images for an object are illustrated in Fig. 7 and the stereo pairs of each were used to extract the primitives and match them to the all database (training set) objects with Algorithm 1.

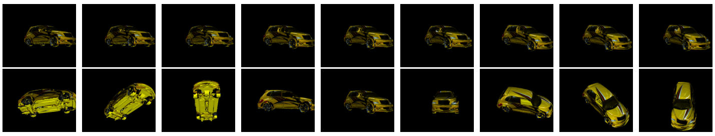


Fig. 7: Variation in the "ToyCarYellow" test images (stereo left): El-Az 5° (top row, the simplest set), and El-Az 40° (bottom, the most difficult set).

Results – The recognition accuracies for all experimental scenarios are presented in Table 3 for the primitive extraction settings Setting 1 and Setting 2 (see Section 3). To compare 2D and 3D matching we utilised directly the 2D primitives with and without the accumulation.

Table 3: Recognition accuracies for the KIT object models (tot. of 1071 test image per set) using median matching (pure chance 0.08%).

Method	*El-Az* 5°	*El-Az* 10°	*El-Az* 20°	*El-Az* 30°	*El-Az* 40°
Med match - Sett. 1	98%	93%	78%	55%	33%
Med match - Sett. 1 (2D)	98%	94%	78%	51%	28%
Med match - Sett. 1 (2D, no acc.)	79%	72%	52%	34%	23%
Med match - Sett. 4	99%	97%	87%	63%	38%
Med match - Shape descr. [42]	88%	75%	47%	33%	19%

Using more primitives achieved by, for example, higher resolution images, is beneficial as the Setting 4 provides the best results. However, the Setting 1 is not significantly worse being much faster (ten seconds vs. minutes in our Matlab implementation). Moreover, the importance of the accumulation process is verified as the 2D matching with accumulated 2D primitives is almost the same to the accumulated 3D matching. 3D primitives are more beneficial with large view point changes where 2D transformation cannot represent the view anymore.

Overall, for small view angle variation (azimuth and elevation $\leq 10^\circ$) our recognition rate is almost perfect and for 20° still almost 90%. The accuracy starts to drop after 20° due to the fact that the test views start containing structures not present in the training view.

To compare our method with other descriptors, we implemented the local shape context, originally proposed for 2D in [43], extended to 3D by Frome et al. [42] and similar to the heuristic approach in [16]. The local shape context corresponds to a histogram of 3D primitives appearing in the vicinity of each primitive. The local shape context is simple and efficient to compute. The bin size was optimised by cross-validation and the results are shown in the last row of Table 3. For KIT objects, the local shape context descriptors are clearly inferior to the colour matching, but still perform well with the smaller angles and are thus promising for applications and imaging conditions where the colour is not informative.

It is noteworthy that since our approach is genuine 3D it also produces the object pose $\boldsymbol{T}$ as a side product. The detected poses are coarse (Fig. 8), but provide good initial guesses for more accurate pose optimisation.

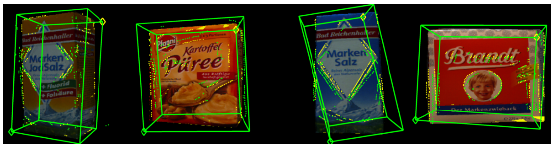


Fig. 8: Extracted 3D primitives (yellow dots) and database object bounding box and 3D primitives (green) projected by the estimated $\boldsymbol{T}$.

5.2 Outdoor street view scenes

In this part of experiment, 160 street-view image pairs at various locations from 4 different cities were used as benchmark database. These database consists of 40 different urban scenes, where each urban scene has 4 street-view pairs, see Fig. 9 **(a)** as an example.

The ground truth camera pose recorded in the metadata of the street-view images were used to estimate approximate camera extrinsics. For each urban scene, we selected one pair of images for training and the rest 3 pairs for testing. Otherwise, all method settings were the same as in the previous experiment. Without any parameter tuning, we achieved satisfactory results as shown in Table 4.

- For 12 classes (or urban scenes) with 48 street-view pairs, the pipeline achieved 92% accuracy, and 97% of the results ranked the correct class within the 5 best candidates produced by the algorithm.
- For 24 classes with 96 street-view pairs, the pipeline achieved 80% accuracy, and 94% of the results ranked the correct class within the 5 best candidates produced by the algorithm.
- For 40 classes with 160 street-view pairs, the pipeline achieved 75% accuracy, and 85% of the results ranked the correct class within the 5 best candidates produced by the algorithm.

The result shows that our 3D promoted primitives and the simple matching algorithm also work with realistic data of moderate occlusion and viewpoint changes.

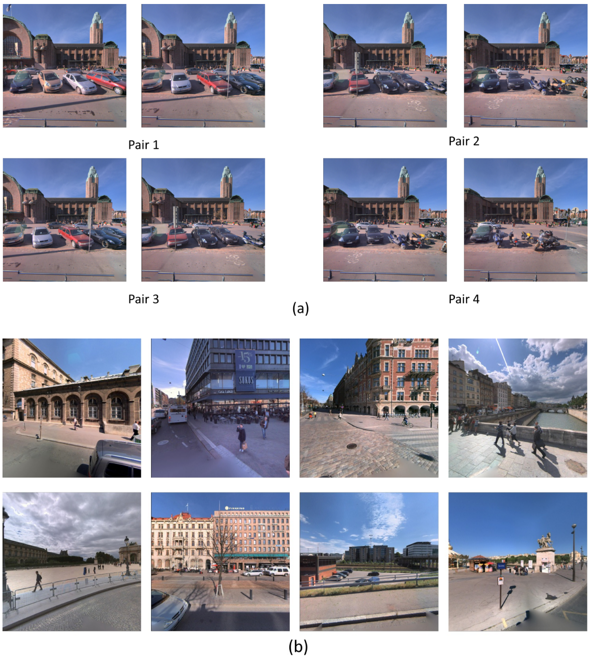


Fig. 9: (a) Here are 4 pairs of street-view images for one urban scene. (b) These are 8 examples of urban scenes from our street-view database.

Table 4: Recognition accuracies for outdoor urban scenes using median matching.

Three Sets	*Set1*	*Set2*	*Set3*
Number of classes	12	24	40
Number of street-view pairs	48	96	160
By pure chance to find the correct class	8%	4%	2%
Accuracy	92%	80%	75%
The correct class within the best 5 candidates	97%	94%	85%

6 Conclusions

This paper proposes an approach that utilizes both the 2D appearance and 3D structure from the multi-view images for 3D object detection and recognition. We introduced novel 3D primitives for indoor objects and urban scenes recognition in 3D. The 3D primitive extraction is based on low level visual 2D primitives selected by computational intrinsic dimension that classifies them according to Marr's primal sketch. The 2D primitives are matched across multi-view images and triangulated to 3D primitives. For matching the primitives, we introduced a simple but effective random sampling procedure that achieved 90% accuracy for the view angle variation up to ±20° with indoor objects dataset and satisfactory accuracy for the street-view dataset. Our future work will include investigation of other primitive types, such as local texture and higher level primitives, such as constant colour regions.

Acknowledgement. The authors would like to give thanks to Dr. Lixin Fan for the valuable discussions.

References

1. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR. (2007)
2. Chum, O., Matas, J.: Unsupervised discovery of co-occurrence in sparse high dimensional data. In: CVPR. (2010)
3. Rodola, E., Albarelli, A., Bergamasco, F., Torsello, A.: A scale independent selection process for 3d object recognition in cluttered scenes. Int J Comput Vis **102** (2013) 129–145
4. As'ari, M., Supriyanto, U.S.E.: 3d shape descriptor for object recognition based on kinect-like depth image. Image and Vision Computing **32** (2014) 260–269
5. Buch, A., Yang, Y., Krüger, N., Petersen, H.: In search of inliers: 3d correspondence by local and global voting. In: CVPR. (2014)
6. Marr, D.: Vision. A Computational Investigation into the Human Representation and Processing of Visual Information. W.H. Freeman and Company (1982)
7. Kalkan, S., Wörgötter, F., Krüger, N.: Statistical analysis of local 3d structure in 2d images. In: CVPR. (2006)

8. Glasner, D., Galun, M., Alpert, S., Basri, R., Shakhnarovich, G.: Viewpoint-aware object detection and pose estimation. In: ICCV. (2011)
9. Sattler, T., Leibe, B., Kobbelt, L.: Fast image-based localization using direct 2d-to-3d matching. In: ICCV. (2011)
10. Zia, M., Stark, M., Schiele, B., Schindler, K.: Detailed 3d representations for object recognition and modeling. IEEE PAMI **35** (2013)
11. Dorai, C., Jain, A.: Shape spectrum based view grouping and matching of 3D free-form objects. T-PAMI **19** (1997)
12. Fayad, J., Russell, C., Agapito, L.: Automated articulated structure and 3D shape recovery from point correspondences. In: ICCV. (2011)
13. Sharma, A., Horaud, R., Cech, J., Boyer, E.: Topologically-robust 3D shape matching based on diffusion geometry and seed growing. In: CVPR. (2011)
14. Bronstein, A., Bronstein, M., Kimmel, R.: Three-dimensional face recognition. Int J Comput Vis **64** (2005)
15. Gökberg, B., Irfanoglu, M., Akarun, L.: 3D shape-based face representation and feature extraction for face recognition. Image and Vision Computing **24** (2006)
16. Papzov, C., Burschka, D.: An efficient RANSAC for 3D object recognition in noisy and occluded scenes. In: ACCV. (2010)
17. Drost, B., Ulrich, M., Navab, N., Ilic, S.: Model globally, match locally: Efficient and robust 3D object recognition. In: CVPR. (2010)
18. Detry, R., Pugeault, N., Piater, J.: A probabilistic framework for 3D visual object representation. T-PAMI **31** (2009)
19. Baseski, E., Pugeault, N., Kalkan, S., Kraft, D., Wörgötter, F., Krüger, N.: A scene representation based on multi-modal 2d and 3d features. In: ICCV Workshop on 3D Representation for Recognition. (2007)
20. Knopp, J., Prasad, M., Willems, G., Timofte, R., van Gool, L.: Hough transform and 3D SURF for robust three dimensional classification. In: ECCV. (2010)
21. Tombari, F., Salti, S., Stefano, L.D.: Unique signatures of histograms for local surface description. In: ECCV. (2010)
22. Pham, M.T., Woodford, O., Perbert, F., Maki, A., Stenger, B., Cipolla, R.: A new distance for scale-invariant 3D shape recognition and registration. In: ICCV. (2011)
23. Zaharescu, A., Boyer, E., Horaud, R.: Keypoints and local descriptors of scalar functions on 2d manifolds. Int J Comput Vis **100** (2012) 78–98
24. Bo, L., Lai, K., Ren, X., Fox, D.: Object recognition with hierarchical kernel descriptors. In: CVPR. (2011)
25. Sun, J., Ovsjanikov, M., Guibas, L.: A concise and provably informative multi-scale signature based on heat diffusion. In: Eurographics Symposium on Geometry Processing. (2009)
26. Bronstein, A., Bronstein, M., Guibas, L., Ovsjanikov, M.: Shape google: Geometric words and expressions for invariant shape retrieval. ACM Trans. on Graphics (2011)
27. Ahmed, N., Theobalt, C., Rössl, C., Thrun, S., Seidel, H.P.: Dense correspondence finding for parameterization-free animation reconstruction from video. In: CVPR. (2008)
28. Mian, A., Bennamoun, M., Owens, R.: On the repeatability and quality of keypoints for local feature-based 3D object retrieval from cluttered scenes. Int J Comput Vis **89** (2010) 348–361
29. Lee, S., Lu, Z., Kim, H.: Probabilistic 3D object recognition with both positive and negative evidences. In: ICCV. (2011)
30. Hu, W., Zhu, S.C.: Learning a probabilistic model mixing 3d and 2d primitives for view invariant object recognition. In: CVPR. (2010)

31. Kang, H., Hebert, M., Kanade, T.: Discovering object instances from scenes of daily living. In: ICCV. (2011)
32. Krüger, N., Janssen, P., Kalkan, S., Lappe, M., Leonardis, A., Piater, J., Rodriguez-Sanchez, A., Wiskott, L.: Deep hierarchies in the primate visual cortex: What can we learn for computer vision? IEEE PAMI **35** (2013)
33. Fidler, S., Boben, M., Leonardis, A.: Similarity-based cross-layered hierarchical representation for object categorization. In: CVPR. (2008)
34. Mutch, J., Lowe, D.: Object class recognition and localization using sparse features with limited receptive fields. Int J Comput Vis **80** (2008) 45–57
35. Pugeault, N., Wörgötter, F., Krüger, N.: Accumulated visual representation for cognitive vision. In: BMVC. (2008)
36. Chaudhuri, B., Sarkar, N.: Texture segmentation using fractal dimension. T-PAMI **17** (1995)
37. Felsberg, M., Sommer, G.: Image features based on a new approach to 2D rotation invariant quadrature filters. In: ECCV. (2002)
38. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. (2003)
39. Chum, O., Matas, J.: Optimal randomized RANSAC. T-PAMI **30** (2008)
40. Umeyama, S.: Least-squares estimation of transformation parameters between two point patterns. T-PAMI **13** (1991)
41. Xue, Z., Kasper, A., Zoellner, J., Dillmann, R.: An automatic grasp planning system for service robots. In: ICAR. (2009)
42. Frome, A., Huber, D., Kolluri, R., Bulow, T., Malik, J.: Recognizing objects in range data using regional point descriptors. In: ECCV. (2004)
43. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape context. T-PAMI **24** (2002)