Camera Re-Localization with Data Augmentation by Image Rendering and Image-to-Image Translation

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Bauingenieur-,
Geo- und Umweltwissenschaften
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

M. Sc. Markus Sebastian Müller

Tag der mündlichen Prüfung:	06. Dezember 2019
Erstgutachter:	PD Dr.-Ing. Boris Jutzi
Zweitgutachter:	Assoc. Prof. Torsten Sattler

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit selbständig verfasst und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe. Die Satzung des Karlsruher Instituts für Technologie (KIT) zur Sicherung guter wissenschaftlicher Praxis wurde in der jeweils gültigen Fassung beachtet.

Karlsruhe, den 24.04.2020 ____________________________

Contents

Acronyms

ANN	Artificial Neural Network.
AR	Augmented Reality.
AS	Active Search.
BRIEF	Binary Robust Independent Elementary Features.
CNN	Convolutional Neural Network.
DoF	Degree of Freedom.
DSAC	Differentiable RANSAC.
DSLR	Digital Single-Lens Reflex.
EPnP	Efficient Perspective-n-Point.
FAST	Features from Accelerated Segment Test.
GAN	Generative Adversarial Network.
GNSS	Global Navigation Satellite System.
LSTM	Long Short-Term Memory.
MAV	Micro Aerial Vehicle.
MLESAC	Maximum Likelihood Estimation Sample Consensus.
MVS	Multi-View Stereo.
NCC	Normalized Cross Correlation.
ORB	Oriented FAST and Rotated BRIEF.
P3P	Perspective-3-Point.

PnP	**Perspective-n-Point.**
RANSAC	**Random Sample Consensus.**
ReLU	**Rectified Linear Unit.**
RGB	**Red-Green-Blue.**
RNN	**Recurrent Neural Network.**
SfM	**Structure from Motion.**
SIFT	**Scale Invariant Feature Transform.**
SLAM	**Simultaneous Localization and Mapping.**
SURF	**Speeded-Up Robust Features.**
TILDE	**Temporally Invariant Learned DEtector.**
UAS	**Unmanned Aerial System.**
UAV	**Unmanned Aerial Vehicle.**
VO	**Visual Odometry.**
VR	**Virtual Reality.**

Abstract

Self-localization of cars, robots or Unmanned Aerial Vehicles (UAVs) as well as self-localization of pedestrians is and will be of high interest for a wide range of applications. A major task is autonomous navigation of vehicles, whereas the localization in the surrounding scene is a key component. Since cameras are well-established built-in sensors in cars, robots and UAVs, there is little to no extra cost in utilizing them for subsequent localization. The same applies for pedestrian localization, where smartphones serve as mobile platforms for cameras. Camera re-localization, where the pose of a camera is determined with respect to a certain scene, is therefore a valuable process to solve or support localization solutions of such vehicles or pedestrians. Cameras are low-cost sensors that are established commonly in the everyday life of humans and machines. The support of camera re-localization is not limited to applications related to navigation but can in general be applied to support image analysis or image processing tasks as scene reconstruction, detection, classification or alike. For these purposes, this thesis concerns the improvement of the camera re-localization task. As Convolutional Neural Networks (CNNs) and hybrid pipelines to regress camera poses are recently competing against established hand-crafted designed pipelines reaching similar or superior accuracies, the focus is set on the former in this thesis. The main contributions of this thesis include the design of a CNN to regress camera poses, with focus on a lightweight architecture fitting the requirements to be applicable on mobile platforms. This network achieves accuracies in the same order as CNNs with larger model sizes. Furthermore, the performance of CNNs is highly depending on the quantity and quality of training data utilized for their optimization. Hence, further contributions are considering image rendering and image-to-image translation to extend such training data in terms of Data Augmentation (DA). 3D models are utilized for image rendering to valuable extend training datasets. Generative Adversarial Networks (GANs) serve for DA by image-to-image translation. Whereas image rendering is increasing the quantity of images in datasets, image-to-image translation on the other hand aims to enhance the quality of such data. Experiments are carried out on datasets augmented by image rendering and image-to-image translation. It is shown, that both approaches valuable enhance the localization concerning accuracy. Therefore, state-of-the-art localization is improved by DA in this thesis.

Kurzfassung

Die Selbstlokalisierung von Automobilen, Robotern oder unbemannten Luftfahrzeugen sowie die Selbstlokalisierung von Fußgängern ist und wird für eine Vielzahl an Anwendungen von hohem Interesse sein. Eine Hauptaufgabe ist die autonome Navigation von solchen Fahrzeugen, wobei die Lokalisierung in der umgebenden Szene eine Schlüsselkomponente darstellt. Da Kameras etablierte fest verbaute Sensoren in Automobilen, Robotern und unbemannten Luftfahrzeugen sind, ist der Mehraufwand diese auch für Aufgaben der Lokalisierung zu verwenden gering bis gar nicht vorhanden. Das gleiche gilt für die Selbstlokalisierung von Fußgängern, bei der Smartphones als mobile Plattformen für Kameras zum Einsatz kommen. Kamera-Relokalisierung, bei der die Pose einer Kamera bezüglich einer festen Umgebung bestimmt wird, ist ein wertvoller Prozess um eine Lösung oder Unterstützung der Lokalisierung für Fahrzeuge oder Fußgänger darzustellen. Kameras sind zudem kostengünstige Sensoren welche im Alltag von Menschen und Maschinen etabliert sind. Die Unterstützung von Kamera-Relokalisierung ist nicht auf Anwendungen bezüglich der Navigation begrenzt, sondern kann allgemein zur Unterstützung von Bildanalyse oder Bildverarbeitung wie Szenenrekonstruktion, Detektion, Klassifizierung oder ähnlichen Anwendungen genutzt werden. Für diese Zwecke, befasst sich diese Arbeit mit der Verbesserung des Prozesses der Kamera-Relokalisierung. Da Convolutional Neural Networks (CNNs) und hybride Lösungen um die Posen von Kameras zu bestimmen in den letzten Jahren mit etablierten manuell entworfenen Methoden konkurrieren, ist der Fokus in dieser Thesis auf erstere Methoden gesetzt. Die Hauptbeiträge dieser Arbeit beinhalten den Entwurf eines CNN zur Schätzung von Kameraposen, wobei der Schwerpunkt auf einer flachen Architektur liegt, die den Anforderungen an mobile Plattformen genügt. Dieses Netzwerk erreicht Genauigkeiten in gleichem Grad wie tiefere CNNs mit umfangreicheren Modelgrößen. Desweiteren ist die Performanz von CNNs stark von der Quantität und Qualität der zugrundeliegenden Trainingsdaten, die für die Optimierung genutzt werden, abhängig. Daher, befassen sich die weiteren Beiträge dieser Thesis mit dem Rendern von Bildern und Bild-zu-Bild Umwandlungen zur Erweiterung solcher Trainingsdaten. Das generelle Erweitern solcher Trainingsdaten wird Data Augmentation (DA) genannt. Für das Rendern von Bildern zur nützlichen Erweiterung von Trainingsdaten werden 3D Modelle genutzt. Generative Adversarial Networks (GANs) dienen zur Bild-zu-Bild Umwandlung. Während das Rendern von Bildern die Quantität in einem Bilddatensatz erhöht, verbessert die Bild-zu-Bild Umwandlung die Qualität dieser gerenderten Daten. Experimente werden sowohl mit erweiterten Datensätzen aus gerenderten Bildern als auch mit umgewandelten Bildern durchgeführt. Beide Ansätze der DA tragen zur Verbesserung der Genauigkeit der Lokalisierung bei. Somit werden in dieser Arbeit Kamera-Relokalisierung mit modernsten Methoden durch DA verbessert.

1 Introduction

Camera re-localization is a crucial problem in fields of autonomous driving, navigation of robots [CN08] or UAVs [MUJ17], pedestrian self-localization, Simultaneous Localization And Mapping (SLAM) [Dav+07; ED06] or Augmented and Virtual Reality (AR and VR) [CKM08; Lyn+15]. Therefore, camera re-localization is deployed on various platforms including cars, robots, Unmanned Aerial Vehicles (UAVs) and underwater vehicles, smartphones or AR and VR devices. It is crucial to know the localization of such platforms in the present scene for applications as navigation or human interaction. Camera re-localization is the estimation of a camera pose given a single image in a known scene. This problem is also referred to as monocular camera re-localization, one shot re-localization or visual re-localization. The *known* scene is generally given by a set of images and their corresponding poses. Since the scene must be known, this task is specified as re-localization. For simplicity, the term *localization* is synonymously used when referring to the camera re-localization problem in this thesis.

Figure 1.1 depicts the full localization pipeline from the original data (left) to a camera pose (right). The figure is structured as follows. The very top row depicts the general data (blue) and methods (green) forming the pipeline. The second row from the top depicts the data and methods more specific, while the methods are divided into hand-crafted (yellow) and data-driven (orange) approaches. The bottom row depicts particular methods exemplary including recent state of the art.

The modules compounding the camera re-localization pipeline can be generally tackled via hand-crafted or data-driven designed approaches. Localization methods based on hand-crafted local image features, such as Scale Invariant Feature Transform (SIFT) [Low04], Speeded-Up Robust Features (SURF) [BTV06], Features from Accelerated Segment Test (FAST) [RD06b], Binary Robust Independent Elementary Features (BRIEF)[Cal+10] or Oriented FAST and Rotated BRIEF (ORB) [Rub+11] estimate camera poses by a known mathematical model. According methods that are based on such features are Active Search (AS) [SLK11; SLK12; SLK16], solvers for the general Perspective-n-Point (PnP) problem [Gao+03], Perspective-3-Point (P3P) problem [Gao+03; QL99; Har+94] or Efficient Perspective-n-Point (EPnP) problem [LMF08], Structure from Motion (SfM) [KD91; Wu13; SF16] or solvers for image retrieval [AZ13; SM97]. Matching 2D features can be approached in a hand-crafted manner by Normalized Cross Correlation (NCC) [YH09] or by data-driven approaches like MatchNet [Han+15].

Data-driven methods that solve camera re-localization in an end-to-end manner are firstly introduced with the publication of PoseNet [KGC15]. The term end-to-end learning is referred to as the process of learning a complex system considering only the input on

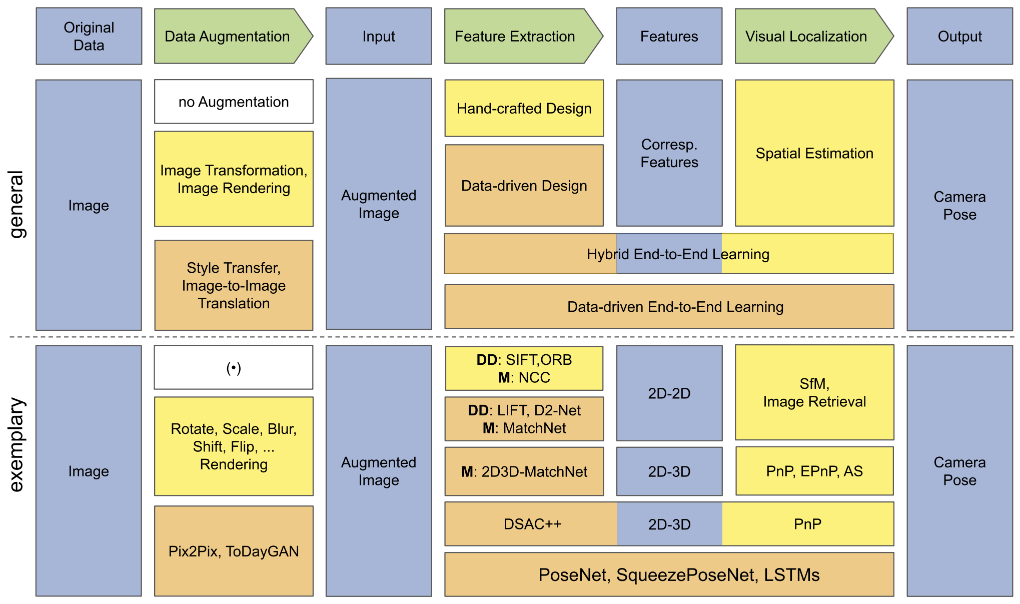


Figure 1.1: Overview of the camera re-localization pipeline. The illustration shows the general image localization pipeline from left (beginning with image data) to the right (ending with a camera pose). The very top row defines data (blue) and methods (green) to solve the localization task. The second row shows the particular data (blue) and general methods based on hand-crafted (yellow) and data-driven (orange) approaches. The bottom row depicts exemplary methods including recent state of the art. Note that the *Feature Extraction* column summarizes detection, description and matching of features. The exemplary methods are marked with **DD** (detection and description) or **M** (matching) depending on what parts of the feature extraction pipeline they solve.

one end and the output on the other end. Convolutional Neural Networks (CNNs) are an example for such networks, that are trained in an end-to-end fashion, whereas the trainable parameters are optimized iteratively without user interaction. Nevertheless, such a model may include non-trainable parts provided in a hand-crafted manner. CNNs are able to estimate a camera pose from single images in a scene after being trained on respective data. This means, that there is no need for hand-crafted designed features between the input (image) and the output (camera pose). The whole model between the input and output is learned during a training process with no or little need for user-based supervision. Enhancements are the Bayesian PoseNet [KC16] which provides a re-localization uncertainty or Long Short-Term Memory (LSTM) [HS97] layers to boost the accuracy of pose estimation [Wal+17]. A lightweight CNN for camera re-localization is introduced with SqueezePoseNet [MUJ17].

Followed by these approaches, that solve the camera re-localization in one step, further methods split up the process of pose estimation in generally two main parts. Feature extraction and localization. As mentioned above, both parts can be tackled by hand-crafted approaches. However, data-driven approaches for feature extraction and localization are established similarly. Feature extraction performed by a CNN like LIFT [Yi+16] outperforms hand-crafted features like SIFT on selected challenging datasets. However, compared on a wide range of scenarios, hand-crafted features perform on similar or better than LIFT [Sch+17]. D2-Net [Dus+19], another CNN, is even capable to produce image features that enable feature matching between images of different domains, like *day* and *night*, *sharp* and *blurred* or between images of different styles. Whereas the most common way is a two-fold process of detection and description, D2-Net handles the entire feature extraction simultaneously.

The feature space of localization approaches like the mentioned SfM utilizes 2D image features like SIFT. The EPnP algorithm on the other hand is based on correspondences of 2D features in an image and 3D features in the world. Such features and their correspondences can be derived by data-driven approaches like 2D3D-MatchNet [Fen+19], a network that jointly learns the descriptors for 2D and 3D keypoints from images and point clouds.

The hand-crafted and data-driven division continues for image retrieval, with hand-crafted approaches [Phi+07; NS06] and data-driven approaches [Gor+16]. Image retrieval is the task of finding the most similar images from a database, given a query image. Therewith, image retrieval does not solve camera re-localization primarily. However, given a database with images and their corresponding poses enables the assignment of a pose to a query image, according to its most similar image(s) from the database. Therefore, image retrieval can be a way to solve re-localization. Besides that, image retrieval is utilized to find relevant parts of 3D models for further 2D-3D point correspondence estimation [Irs+09].

A hybrid method that consists of a CNN for scene coordinate regression and a PnP solver is trained in an end-to-end manner by providing a fully differential pipeline [Bra+17]. This workflow is considered as hybrid since it combines data-driven learning with hand-crafted designed spatial estimation in an end-to-end trained pipeline.

The camera re-localization workflow is completed with approaches considering DA. DA is an optional step, yet very powerful and wide-spread in the computer vision community supporting various tasks. In terms of camera re-localization, DA can be carried out by trivial image transformations such as rotating, scaling, blurring or alike. Ahead on this, image rendering is a valuable process, generating synthetic data from 3D models. Besides these hand-crafted methods, style transfer or image-to-image translation constitute approaches based on data-driven learning. Image rendering and image-to-image translation is covered in this thesis to support camera re-localization.

1.1 Motivation

This thesis aims on the investigation and improvement of camera re-localization and is motivated by the rising interest of disciplines like computer vision and robotics deploying such methods for solving initial localization in Visual Odometry (VO) and SLAM-based systems, for complementing existing localization frameworks or for usage as deployable stand-alone systems. Such re-localization is a necessity in the fields of autonomous driving or AR and VR. In contrast to local positioning techniques as VO or SLAM which derive camera poses in a local reference frame, camera re-localization derives camera poses in a global reference frame, assuming accordingly labeled training data. This is more valuable given informations about the absolute position and orientation in a global reference system. If no labeled image poses in a global reference frame are provided, camera re-localization can only carried out locally with respect to a scene coordinate frame.

Applications for the navigation of vehicles or the self-localization of pedestrians rely on real-time or near real-time processing to be applicable in real-world scenarios. Data-driven deep learning methods are known for their excessive time consuming training processes including the so-called backward propagation – or short backpropagation – relying on sufficient hardware to be practicable utilized. However, during runtime only forward passes need to be handled, which are basically chained multiplications that can be processed within milliseconds.

Localizing vehicles or robots is often carried out by utilizing Global Navigation Satellite Systems (GNSSs). As camera re-localization, GNSS methods derive a pose in a global reference frame. However, GNSS generally lacks accuracy from shadowing and multi-path effects or fails in indoor or strong occluded scenes. Camera re-localization overcomes these issues and therefore effectively complements existing GNSS localization. Camera re-localization differs from GNSS in a way that they are both inside-out localization approaches, whereas a sensor gathers information about the surrounding scene to localize itself within this scene. Considering GNSS, the scene has to be equipped with satellites providing active signals (space segment). A rover determines its position processing these received signals. An orientation can not be determined by using merely GNSS. Also the user does not have assured control of the so-called space segment. In contrast, camera re-localization relies merely on images captured in a scene or a 3D model and provide camera orientations as well.

The number of images publicly available or owned by companies (like *Google LCC* or *flickr*) increased throughout the last years. Important landmarks and parts of cities could successfully be reconstructed utilizing only publicly available images from the internet. Projects like *BigSFM: Reconstructing the World from Internet Photos* [Uni] or *Reconstructing the World in Six Days* [Hei+15] have the objective to reconstruct the whole world or large parts of it using only internet photo collections. Therewith, one of the modules for camera re-localization - the data - is available in a vast amount.

1.2 Problems and Contribution

With the success of deep learning, the problem of camera re-localization experienced an essential change concerning methodological solvers. While the problem is well known and a popular field of research improved by hand-crafted advancements over the last years, data-driven methods caught the interest of many researches in the field. While the first data-driven methods could solve camera re-localization only up to a certain degree not outperforming existing methods based on hand-crafted design, the data-driven approaches improved over time and caught up on hand-crafted methods or even surpassing them lately for particular scenes.

In the following, existing problems and the corresponding contributions of this thesis tackling these problems are highlighted.

- **Problem I: Limitation of memory capacity**. Navigation frameworks benefit from camera re-localization. Potential platforms for such navigation frameworks are cars, robots, UAVs, unmanned underwater vehicles, smartphones or alike. A drawback of data-driven deep learning methods is their system requirements. Due to the high number of parameters stored in CNN models, the memory capacity of mobile devices or embedded computers attached to such platforms is a limiting factor. When large CNNs do not satisfy the requirements for on-board processing, lightweight networks could potentially substitute them without simultaneous drawbacks.
  Contribution I: A lightweight CNN [MUJ17] to solve camera re-localization by pose regression is proposed (Chapter 5). This network is capable of running on embedded computers, deployable on UAVs or small robots.

- **Problem II: Lack of training data**. Camera re-localization requires distinct knowledge of a regarding scene to be functional. This knowledge is at least given by images covering the scene and optionally their corresponding image poses or by 3D models. Considering CNNs for camera re-localization, the training data usually consists of images and their corresponding poses. The performance of such networks depends heavily on the underlying training data. The amount of training data and its distribution affect the performance significant, whereas networks generally benefit from a high amount of training data and a rich distribution of such. If such training data is not provided with sufficient quantity and distribution, the camera re-localization suffers or fails.
  Contribution II: Data Augmentation (DA) by image rendering is proposed [MJ18; MMJ18] to overcome the stated problem of lacking training data (Chapter 6). Therefore, images are rendered from 3D models to generate an arbitrary amount of training data while increasing its distribution.

- **Problem III: Domain differences**. As stated in the description of Problem II, the issue of lacking training data is overcome by DA by utilizing image rendering. The training data is either enhanced by rendered images or exclusively consists of such. These rendered images appear – depending on the 3D model serving for

rendering – with visual appearance or styles that differ from the characteristics of the evaluation data. Therewith, training and evaluation images exist in different domains. Training a network on a specific domain X and operating on domain Y leads to inconsistencies, since the network has never learned about the distribution of domain Y. This issue leads to dissimilarities in the feature space and affects the camera pose estimation negatively.
Contribution III: Image-to-image translation is utilized for mapping images between different domains [Mül+19] to overcome recent drawbacks (Chapter 7). By image-to-image translation the radiometric and geometric information of images is mapped to a target domain improving the characteristics considering the similarity of training and evaluation data. Generative Adversarial Networks (GANs) are trained for the mapping between such image domains. Therewith, the accuracy of camera re-localization is enhanced.

1.3 Organization of the Thesis

The thesis is structured as follows. Chapter 1 introduces camera re-localization, motivates the relevance of the topic and outlines the contributions. Chapter 2 summarizes important works and contributions to the camera re-localization problem considering DA, feature extraction and visual localization. The fundamentals of deep learning, including CNNs and GANs are outlined in Chapter 3 as such methods are utilized throughout the thesis. Important datasets utilized or acquired within this thesis are outlined in Chapter 4. The three chapters considering localization without DA (Chapter 5), localization with DA by image rendering (Chapter 6) and localization with DA by image-to-image translation (Chapter 7) form the methodological main chapters. Subsequently, the corresponding experiments are carried out in Chapter 8. The results of the experiments are discussed in Chapter 9. Finally a conclusion and outlook of the thesis is presented with Chapter 10.

2 State of the Art on Camera Re-Localization

This chapter presents an overview of the modules compounding the camera re-localization pipeline from raw image data over optional DA to feature extraction and finally the localization process to derive an image pose. The definition of an image pose and its parametrization is conducted in the Appendix A. The general aim is to estimate an image pose with respect to a fixed reference frame given a set of images and their poses in the same reference frame. Such an image pose can be estimated by different localization methods (Section 2.3), either by spatial estimation modeled in a hand-crafted manner (Section 2.3.1) or by end-to-end learned models (Section 2.3.2). Both strategies rely in some way on features, extracted from the input images. The approaches to extract such features can be split into hand-crafted and data-driven methods and are described in Section 2.2. The input images for such feature extraction can be represented by the original data or by augmented data originated from the former. Such DA is covered in Section 2.1 and can be conducted by either transformations of the original images or by generating completely new images by image rendering or image-to-image translation. All these approaches use merely the original data as input, either in a direct or indirect way.

The sections compounding the fundamentals are tightly coupled to the camera re-localization pipeline shown in Figure 1.1. As illustrated, the three methodological modules *Data Augmentation*, *Feature Extraction* and *Visual Localization* (green) each can be approached by either *hand-crafted* (yellow) or *data-driven* (orange) solutions. Therefore, the sections of fundamentals are loosely split into subsections focusing on both hand-crafted and data-driven methodology. Within the pipeline of determining an image pose, the sub-methods of localization, feature extraction and DA can be modularly exchanged partially, whereas the decision which parts should be carried out by hand-crafted methods and which parts by data-driven methods to derive best results for an image pose is of high interest. In this thesis, selected modules from the image localization pipeline are enhanced to advance the state of the art.

2.1 Data Augmentation

DA is a scheme that artificially inflates a dataset by generating additional, type preserving data using domain specific synthesization. Data-driven and hand-crafted methods rely on sufficient training data to maximize performance. With the limitation of such data, both these approaches run into deficiency. These problems occur and are maximized with the level of discrepancy between training and evaluation data. Generally, the more similarity a evaluation dataset shares with its linked training dataset, the more satisfactory can a subsequent task be performed. Such similarity can not always be ensured and is often not favored in terms of transferability and generalization. If a pipeline is aimed to provide a preferable general and robust performance on arbitrary data, it is not desirable to provide a training set with similar distribution of data, since this leads to overfitting. Overfitting is referred to systems or models when they perform satisfactory on a particular data but not on general data. These models are therefore overfitted to a restricted number of appearances but can not be deployed on more general tasks. Aside from gathering more instances of data samples that are representative for a distinct task, the process of DA refers to the operation to generate more samples by synthesization given a basis of data. In the case of image pose estimation, the robustness and accuracy of an applied method highly relies on the basis of the given image data. The performance of both, data-driven and hand-crafted localization methods is improved by DA [TN17; PW17; MJ18; MMJ18; Mül+19]. Considering the pipeline in Figure 1.1, DA is the module that impacts the pose estimation throughout the whole processing chain, independent of the choice of subsequent processing steps. It fundamentally impacts both, hand-crafted and data-driven methods and is therefore an affective module of interest in the processing chain. Lack of training data is a major problem in different fields of computer vision and other disciplines. The generation of training data is often tedious and expensive due to manual annotations and labeling. DA generally provides a beneficial option to generate additional data with little cost. DA includes image transformations, image rendering and image-to-image translation. Image transformations (Section 2.1.1) include generic and often trivial geometric and radiometric transformations that are applied to base images to directly produce variants of new images. Image rendering (Section 2.1.2) is carried out by utilizing a 3D model of a scene to generate an arbitrary amount of images from synthetic poses. The advantage of this method is, that images with purely new scene views and poses are added to the database that provide beneficial value. Image-to-image translation introduced in Section 7.3.1 is the process of mapping an input image in a certain source domain to an output image in a target domain. At the frontier for such image-to-image translation are GANs [Goo+14], that are able to generate synthetic images in particular domains (like summer and winter) or with artistically styles.

2.1.1 Image Transformation

A trivial way to expand an image dataset is to artificially enlarge the dataset by label-preserving radiometric and geometric image transformations. Common geometric and radiometric transformations include but are not limited to scaling, rotating, shifting, smoothing, cropping, edge enhancement, blurring, regional dropout, color jittering, adding different types of noise and affine, perspective or radiometric transformations or principal component analysis. This supports algorithms to learn invariance, to e.g. shift and rotation, which helps to generalize and increase the accuracy of the model. Figure 2.1 illustrates different kinds of image transformations considering a single input image (left). The sheer size of a training dataset can be increased significantly with such exemplary transformations. Applications like classification [He+15; How13; KSH12], object detection [He+15], action recognition in videos [SZ14a] or handwriting recognition [SSP03] benefit from this technique. The authors of the latter publication state, that getting the training set as large as possible is "the most important practice". In general image transformations support data-driven methods to achieve invariance and/or robustness to geometry and radiometry. Moreover, by varying the transformation parameters, an infinite amount of augmented data may be generated.

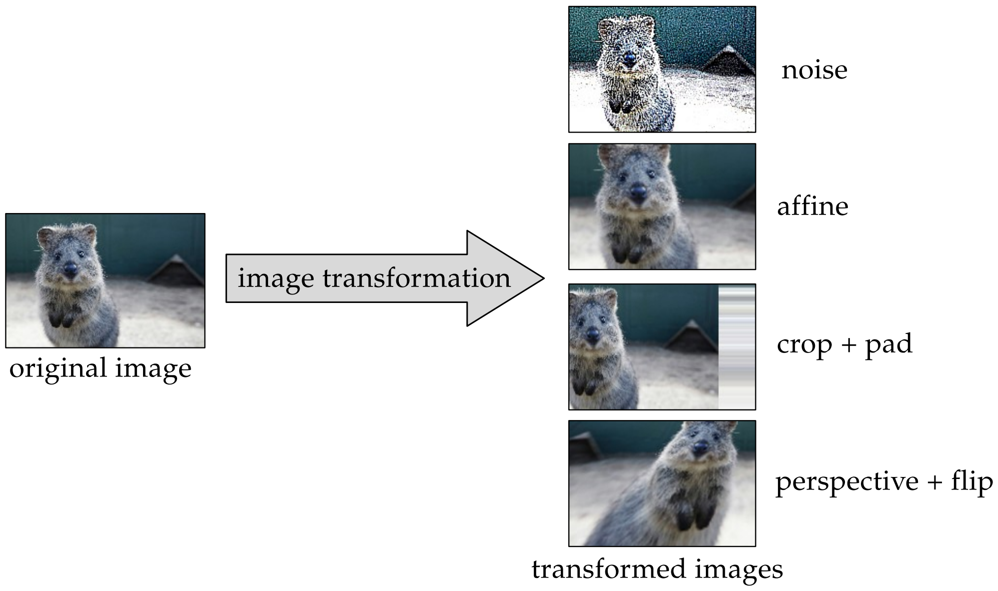


Figure 2.1: Examples for image transformations [Jun+19]. An original image is transformed by various single mapping functions or combinations of them. Further transformations could be blur, rotate, regional dropout, colour jittering, channel shuffle, gray scale conversion and other geometric or radiometric transformations.

2.1.2 Image Rendering

In contrast to the previous outlined method on image transformation for DA, image rendering is a more complex but also a more versatile way to enhance an existing image dataset. Image rendering is the process of generating completely new data from artificial or reconstructed 3D models from an arbitrary camera viewpoint respectively an arbitrary pose. Figure 2.2 depicts this process. Rendering can potentially provide infinite and generally richer training data compared to the generic augmentation methods mentioned in Section 2.1.1. DA by image rendering is successfully utilized for hand gesture recognition [LA17], face recognition [Mas+19] or viewpoint estimation [Su+15]. 3D models are also used to learn deep object detectors [Pen+15]. Images of faces from purely new viewpoints are rendered to provide richer data for face recognition [Lom+18]. The new views are generated given only an original image of a face. In general, image transformations (Section 2.1.1) on rendered images can also be utilized to further enhance a training dataset.

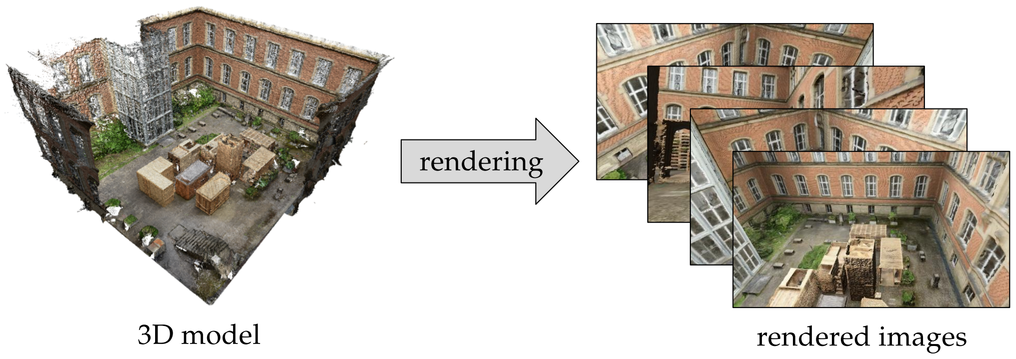


Figure 2.2: Overview of the image rendering scheme. Given a 3D model, any number of images can be rendered from arbitrary poses. These new images can then be added to a training dataset and be utilized to enhance methods for camera re-localization.

2.1.3 Image-to-Image Translation

Image-to-image translation is the process of mapping an input image in domain y to an output image in domain x (Figure 2.3). Such translations can be applied to map images between different artistically styles, seasonal domains or generally for image enhancements. The domains are conditioned from the distribution of image characteristics. Exemplary domains for such mappings are *spring*, *summer*, *autumn* and *winter* [Zhu+17], *day* and *night* [Ano+19], *rendered* and *captured* [Mül+19] or translations between different artistically styles [Ano+18; Zhu+17]. By image-to-image translation, images can be mapped between these domains, generally in all directions. A summer image can be translated into a winter image or vice-versa. This process is recently carried out by GANs which are described more detailed in Section 3.3. Rendered images of eyes are translated to reduce the gap to distributions of real images to improve training on synthetic images [Shr+17b]. Image-to-image translation made a huge leap benefiting from the advancements and better understanding of deep learning and could provide beneficial value serving for DA. The idea is to learn a mapping between two domains whereas the target domain coincides with the image characteristics of a training dataset. From image rendering, images are generated with a deficient quality concerning conformity to the training dataset. By image-to-image translation, this issue can be overcome by mapping the rendered images from their rendered domain to images in a captured domain increasing the quality of such images considering their appearance characteristics. Key differences of the mentioned image-to-image translation approaches are that they are based on GANs that do not need paired training data. Concerning a mapping between summer images and winter images, there is no need to provide a labeled image pair in both domains for training. Rather the algorithms are able to learn solely from arbitrary, un-paired summer and winter images. This is a huge advantage, since the tedious labeling or the sheer collection of such data is often hard or even impossible to achieve.

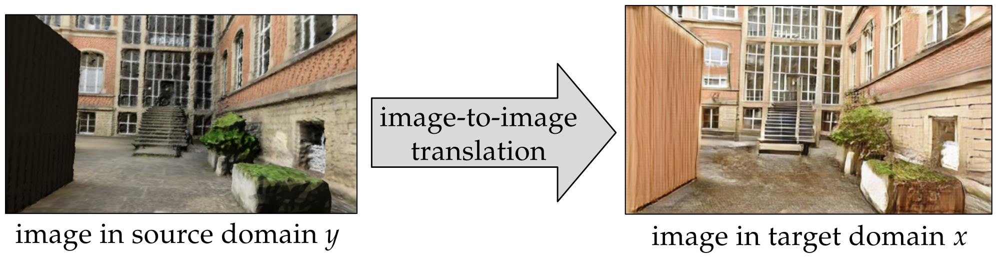


image-to-image translation
image in source domain y
image in target domain x

Figure 2.3: Exemplary image-to-image translation. An image in a source domain y is mapped into an image in the target domain x. The image-to-image translation shown in this figure is carried out from a rendered image in the *rendered* domain y to a translated image in the *captured* domain x.

2.2 Feature Extraction

Feature extraction is a two-fold task that is composed by detecting distinct points or interest points (feature detection) and describing them (feature description). Feature detection and description can be solved by either hand-crafted methods which are focused in Section 2.2.1 or data-driven methods which are focused in Section 2.2.2. Feature extraction is a key function in computer vision and photogrammetry as being a fundamental module that is mandatory for numerous higher level tasks in this fields. It is also a prerequisite for all in Section 2.3.1 mentioned spatial estimation methods. Data-driven end-to-end methods also utilize features to estimate poses. However, these methods learn suitable features within the learning pipeline.

2.2.1 Hand-crafted Design

Hand-crafted feature extraction is the task to detect and describe features at points of interest or at keypoints in an image by hand-crafted designed image analysis methods. A feature should represent a distinctive and unique part of an image. Such features are used as a starting point in many computer vision and photogrammetry algorithms including object tracking, image matching or classification. Feature extraction is an initial operation for many algorithms and is applied directly on raw or preprocessed data. The repeatability of a feature is of high importance to detect corresponding features reliable over two or more images with varying viewpoints or in varying domains.

The initial part of feature extraction is feature detection. Many algorithms for feature detection have been proposed, which vary widely in the kinds of features that are detected, the computational complexity and the repeatability. Such algorithms generally detect points as corners or blobs. The first feature detectors are introduced by Harris and Stevens [HS88] finding interest points by calculating differences of intensity. Shi and Tomasi [ST94] modified that detector improving the performance by changing the score function that decides if a corner is existent in a patch or not. Other corner detectors are BRIEF, ORB, Adaptive and Generic corner detection based on the Accelerated Segment Test (AGAST) [Mai+10] or Binary Robust Invariant Scalable Keypoints (BRISK) [LCS11]. The most popular blob detectors are SIFT and the more efficient variant SURF. The SIFT detector computes Gaussian filters with different variances and detects extrema in Difference of Gaussians computed over neighboring filtered maps creating a scale space. With SURF this process is sped up by approximating the Gaussian derivatives with box filters and utilizing integral images, therefore upscaling the filter size does increase computational complexity. The KAZE feature detector avoids blurring the images by non-linear diffusion [ABD12] therefore avoids smoothing details of present objects. KAZE reaches a higher repeatability over SIFT. These algorithms examine every pixel in an image and decide if there is an *interesting* region around that pixel or not.

The second part of feature extraction is feature description. After finding distinct points, description for these points based on their local neighborhood are computed. The description of an image feature is usually mapped into a descriptor. If a detector provides information about the orientation or scale of a feature, the descriptor can be made invariant to such transformations. Descriptors are stored as floating point values or binary strings. These descriptors are then utilized to e.g. compare image contents, perform image matching or compute image transformations.

However, feature extraction is often one of the more computationally expensive parts of an image processing pipeline. Learned methods aim to reduce the expenditure of time, while generating more robust and distinct features in varying conditions.

2.2.2 Data-driven Design

In contrast to hand-crafted feature extraction, pipelines that detect and describe features can be alternatively learned in a data-driven manner. Approaches learn such pipelines for feature extraction in an end-to-end manner. End-to-end learning is the process of learning all parameters of a model jointly rather than in multiple single steps. Therefore, a model is optimized given input and desired output, whereas no parameters have to be set manually during the training process. To train such pipelines in an end-to-end fashion by deep learning methods, the differentiability has to be preserved, which can be one of the most difficult problems of such tasks. Approaches solve that problem for single parts of the feature extraction pipeline separately. A feature detector that extracts repeatable keypoints under drastic image changes like varying weather conditions or lighting changes from day to night is given with the Temporally Invariant Learned DEtector (TILDE) [Ver+15]. TILDE outperforms hand-crafted features like SIFT, SURF, FAST and others on challenging datasets. A feature orientation extractor learned by a siamese CNN [Moo+16] improves the matching of feature points over orientations extracted from SIFT, SURF, FAST, ORB, KAZE and others. The approach also leads to enhanced Multi-View Stereo (MVS) results improving the completeness and the details of generated point clouds. A feature descriptor learned by a siamese CNN [Sim+15] shows a higher robustness to rotation than SIFT and a data-driven approach [SVZ14]. The authors state that the descriptor generalizes well against scaling, rotation, perspective transformations, non-rigid deformation and illumination changes. The descriptor has a 128-dimensionality and can therefore easily be used as a replacement for descriptors like SIFT which has the same dimensionality. Further developments aim to solve more parts of the feature extraction pipeline at once.

Feature detection with a jointly computation of feature similarities for subsequent feature matching is introduced with MatchNet [Han+15]. To ensure higher repeatability, the training data is augmented with synthetic samples that help reduce overfitting. An interesting part is the study on the trade-off between feature dimension (storage) and accuracy. The accuracy improves with increasing feature dimension. The paper shows that CNNs are effective for wide-baseline patch matching. MatchNet produces superior accuracies compared to SIFT and a data-driven approach [SVZ14] using a 4096-dimension feature. Feature dimensionalities (64, 128, 512) trained with MatchNet score similar or better than other hand-crafted and data-driven matching approaches. The first pipeline that solves feature detection, feature orientation estimation and feature description jointly preserving end-to-end differentiability is the Learned Invariant Feature Transform (LIFT) [Yi+16]. Spatial Transformers [Jad+15] are used to consecutively assemble the individual steps of feature detection [Ver+15], feature orientation [Moo+16] and feature description [Sim+15]. The spatial transformer is utilized to rectify the output image patches of the detector and the orientation estimator to serve as an input for the subsequent modules. Furthermore a soft *arg max* function is applied for non-local maxima suppression while preserving differentiability. Another combined solver for feature detection and description is introduced with D2-Net [Dus+19]. The network based on a single CNN solves the feature extraction task in its full extent. It outperforms variants of SIFT as well as other data-driven approaches under difficult imaging conditions and appearance changes such as illumination differences from daytime to nighttime, motion blur or even changes in the artistically style of the images. The key idea is to address the detection step in a later stage within the network rather than on low-level information basis in the early stages, thus leading to a higher stability. Such feature detection pipelines outperform hand-crafted methods lately for datasets with challenging conditions as the Aachen Day-Night dataset [Sat+18a].

An other approach for image pose estimation is utilizing 2D-3D matches. Besides the 2D features extracted from images, 3D information is necessarily provided or generated within the process. By finding 2D-3D correspondences, camera localization can be solved – for instance by a Perspective-n-Point (PnP) solver. 3D point clouds store valuable information to solve or support camera localization. A drawback is the difficulty in obtaining 2D-3D image to point cloud correspondences. An approach that focuses on finding such correspondences reliable is the 2D3D-MatchNet [Fen+19], an end-to-end trainable network that jointly learns descriptors for 2D and 3D points in images respectively point clouds, whereas a point cloud of the scene has to be provided.

A further approach focuses on generating 3D points of the scene on the fly by learning a scene coordinate regression CNN and a scoring hypothesis function for camera poses from 2D-3D correspondences generated by the former scene regression. To learn such a pipeline in an end-to-end manner all parts have to be differentiable. The usually used RANSAC algorithm [FB81], whose hypothesis selection is not differentiable is substituted by the Differentiable RANSAC (DSAC). DSAC learns scene coordinate regression and pose hypothesis scoring jointly [BR18; Bra+17]. However, the only learnable component of the pipeline is the scene regression CNN, generating 2D-3D correspondences. Subsequently, a PnP solver determines the camera localizations by a known mathematical model in a hand-crafted manner. Scene coordinate regressors are originally based on Random Forest approaches [Sho+13a; Bra+16].

A further 2D-3D approach is the Descriptor-Matcher [Nad+19], which exploits 3D features extracted from dense point clouds as well as 2D features extracted from images. This 2D-3D descriptor matcher is based on a Random Forest [Bre01] classifier. The matcher is trained on 2D SIFT- and 3D descriptors such as 3D-SIFT [SAS07] or Harris 3D [SB11]. Subsequently, the derived 2D-3D matches are used to solve the P3P problem with the robust Maximum Likelihood Estimation Sample Consensus (MLESAC) [TZ00] estimator. Their experiments show competitive results compared to hand-crafted methods and methods that learn 2D-3D correspondences utilizing a CNN.

2.3 Visual Localization

In this section an overview of visual localization methods is presented. The localization task is also known as re-localization, since prior knowledge of the scene has to be given. Generally visual localization is the process of determining the pose of an image with respect to a given reference frame in six Degrees of Freedom (DoF). In contrast to localization but close related is the relative poses estimation between one or more images as in VO or SLAM systems. A key difference is the determination of the pose in a global reference frame by localization methods in contrast to the determination of a pose with respect to a local reference frame (usually a neighboring image or an initial keyframe) by relative pose estimation. Considering Figure 1.1, visual localization is split into methods of spatial estimation and end-to-end learning. Methods of these divisions are subsequently introduced in Section 2.3.1 and Section 2.3.2.

2.3.1 Spatial Estimation

In this section a brief introduction to spatial estimation that utilize hand-crafted designed algorithms to derive an image pose is given. State-of-the-art solutions are solvers for the PnP problem, SfM and image retrieval, which are relevant for this thesis and will be highlighted in the following.

The **Perspective-n-Point** problem is the issue of estimating the pose of a camera given a set of n 3D points in a scene and their corresponding 2D points in an image. The camera pose consists of six DoF which are composed by the orientation (*roll*, *pitch*, and *yaw*) and the position (x, y, z) of the camera with respect to a higher-level reference frame. The image pose of a camera with center $\mathbf{C}_0$ is given by a rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ with respect to a global reference frame $\mathbf{W}_0$. The image pose is computed from corresponding 2D points and 3D points (Figure 2.4).

PnP is well known in the computer vision [HZ03] and photogrammetry [MMB04] communities. A commonly used solution to the problem is P3P [Gru41; Har+94], where

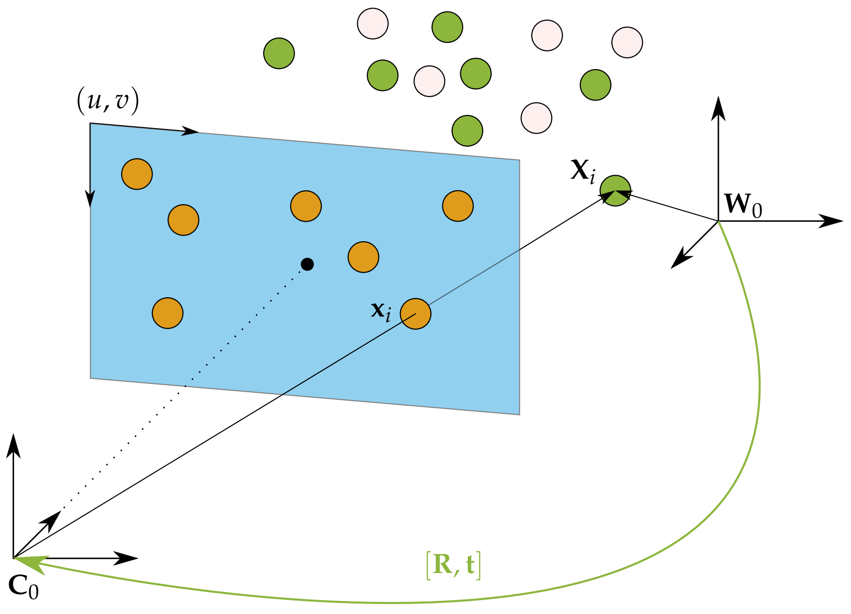


Figure 2.4: The Perspective-n-Point problem is stated as the issue of estimating an image pose given a set of 3D points of a scene and their corresponding 2D points in an image. The image pose of a camera with center $\mathbf{C}_0$ is given by a rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ with respect to a global reference frame $\mathbf{W}_0$. The image pose is computed from corresponding 2D points (orange) and 3D points (green).

$n = 3$ and many solutions are available for the general case of $n > 3$. A solution for $n = 2$ only exists when additional information, like feature orientations at the two particular points, is given. A more common case would be a known camera direction that is acquired by inertial measurements or by constructed vanishing points in images [KBP10]. EPnP [LMF08] is a non-iterative solution for $n \geq 4$ with a complexity linear growing with n. In contrast, other works on this problem have complexities of $O(n^5)$ or up to $O(n^8)$ at same accuracies. With EPnP, each of the n points are expressed as a weighted sum of four virtual control points. The unknown parameters of the problem are then the coordinates of the control points which serve for solving the camera pose. This reduces the problem to estimating the coordinates of these points in the camera to $O(n)$. The first non-iterative PnP solution that achieves better results than iterative solutions concerning accuracy is the Robust PnP [LXX12]. Further advantages are the handling of planar cases, ordinary 3D cases and quasi-singular cases while achieving a computational complexity of $O(n)$.

An n-point algorithm is used for camera pose estimation in combination with AS, where 2D-3D correspondences between images and a 3D model are found by firstly using a SIFT ratio test to reject ambiguous matches and secondly solving an n-point algorithm inside a RANSAC loop. The key novelty of the AS approach is the *active correspondence search*, where correspondences are actively searched in a region where a 2D-3D match is already found. While achieving comparable or superior runtime against other localization methods. AS scores the best registration performance on three benchmark datasets of, which are Vienna [Irs+09], Rome and Dubrovnik [LSH10].

Structure from Motion is the process of simultaneous 3D surface reconstruction and image pose estimation [Ull79]. Given multiple images showing different but partly overlapping views of a particular scene, SfM aims to reconstruct the scene while simultaneously estimating the camera poses with respect to the scene (Figure 2.5). Initially, the relative

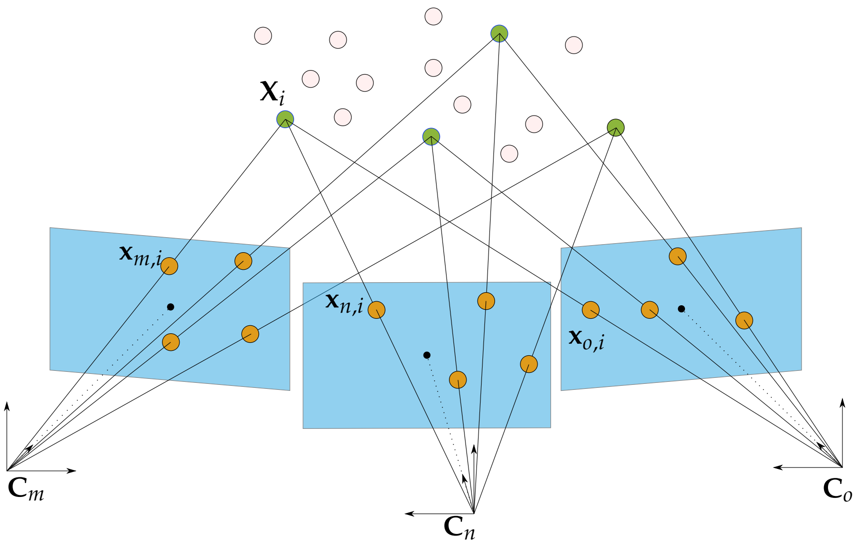


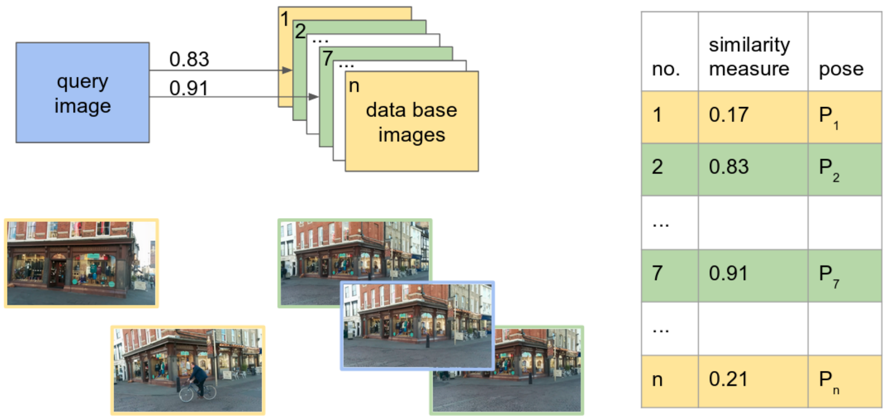
Figure 2.5: Given a number of images with overlapping scene views, SfM is the task of reconstructing 3D points of the scene while estimating the relative image poses.

orientations and translations between the utilized images have to be computed. This is also known as image alignment or image registration. A necessity for that is the feature extraction (Section 2.2) of distinctive points in single images and the matching of homologous features in view-overlapping images. This is generally followed by a robust estimator for outlier detection like RANSAC and subsequently completed by Bundle Adjustment [Tri+99].

Bundle Adjustment is the process of simultaneously optimizing a surface reconstruction expressed as 3D points as well as camera poses and optionally the cameras internal parameters. This optimization aims to minimize the reprojection error between the reprojected scene points in an image and the corresponding measured points in the same image. The minimization is processed using nonlinear least-square algorithms. The runtime of such optimizations scales with the number of images and can lead to time-consuming procedures.

Early self-calibrating systems [MQV95; BTZ96; FZ98] are followed by nowadays well-established SfM pipelines, which are provided with VisualSFM [Wu13] or COLMAP [SF16; Sch+16]. Pipelines like this are often used to generate labeled training data for camera re-localization or related tasks.

Image retrieval is the task of searching and retrieving images from a data base of multiple images. This could be solved with so-called meta search, where the images have associated meta data such as text, keywords or labels if such are existent. Such meta data is not generally provided and only raw images are stored in data bases. Therefore, only the images content composed of its pixel data can be used for a retrieval task. One image retrieval method, based on such data is Content Based Image Retrieval (CBIR), where colors, shapes or textures of an image are analyzed by computer vision algorithms to find similarities between two or more images. Given a query image, the task is to retrieve the most similar image(s) from the database. Generally this is unrelated to pose estimation. However, under the assumption that similar images have similar poses, one can determine poses of query images if the poses of the data base images are provided. Thus, by image retrieval the pose of an image is determined according to a measure of similarity to one or multiple other image(s) in a data base. Let I_q be the query image and I_{db}^n a number of n data base images with known poses. By determining a measure of similarity one can assign the most similar image(s) in a data base to a query image. The pose of a query image can then be estimated by simply assigning the pose of the most similar data base image to the query image. Likewise one could assign a median or mean value of the m most similar data base images to the query image. Figure 2.6 illustrates the general approach of image retrieval. Early successful retrieval systems query images with respect to image data bases



no.	similarity measure	pose
1	0.17	P_1
2	0.83	P_2
...		
7	0.91	P_7
...		
n	0.21	P_n

Figure 2.6: Image Retrieval. A query image is compared to data base images by a similarity measure (e.g. differences of histograms). The pose of the query image can then be estimated by assigning the pose of the most similar image (according to the similarity measure) or a mean value of the nearest m images. A geometric scheme is illustrated exemplarly in this figure where a query image (blue) and the two most similar data base images (green) as well as other data base images (yellow) that are less similar to the query image are shown. The pose of the query image is then interpolated considering the poses of the nearest data base images.

or videos [Fli+95; RHM97].

2.3.2 End-to-End Learning

In this section an introduction to approaches based end-to-end learning to solve visual localization is given. The first deep learning approach that could solve such image localization is presented with the publication of PoseNet [KGC15], a CNN, that is able to regress poses from images in six DoF in a certain scene after being shown a number of images of the same scene and their corresponding poses. Further developments in this field utilize LSTM to boost the accuracy [Wal+17]. These networks generally have no need for additional hand-crafted or manual optimizations. In this section, PoseNet and an LSTM approach will be highlighted as two end-to-end pipelines for image pose regression.

PoseNet is the first CNN to regress image poses in an end-to-end manner and marks a milestone of camera re-localization. The input of the network during training are Red-Green-Blue (RGB) images of a particular scene and the corresponding poses for these images in six DoF. The model is trained in an end-to-end manner, that means that the models parameters are optimized given input, desired output and a loss function. The network is shown to operate indoors and outdoors with a processing time of five milliseconds per image - which is considered as real time for many real-world applications. The accuracy of pose regression on benchmark datasets scores about 2 m and 3° for large outdoor scenes and about 0.5 m and 5° for indoor scenes. The network learns the appearance of the images and their location and is able to regress poses of unseen images of the same scene during test time. This is achieved by a 23 layer deep convolutional network containing convolutional layers, pooling layers, fully connected layers, softmax layers and Rectified Linear Units (ReLUs) for non-linearity. PoseNet shows promising results concerning challenging image characteristics as changing lighting conditions, variable camera intrinsics and motion blur where feature-based SIFT registration fails. The authors justify this robustness by the network's focus on high level features which they demonstrate with the illustration of activation maps. The generation of training data for re-localization networks is simply generated from images datasets. The images are processed through SfM or an equivalent pipeline to obtain a six DoF pose label for every image. This labeled data suffices as training data for PoseNet or other localization networks. PoseNet learns feature vectors, that are mapped to a pose which is stated by the authors to generalize to unseen scenes with only a few additional training samples. However, this conclusion is vague and does not generally hold for any kind of scene without a potential loss of accuracy in the pose estimation. The loss $\mathcal{L}$ of an image I is computed by the loss function

$$\mathcal{L}(I) = \|\hat{x} - x\|_2 + \beta \|\hat{q} - \frac{q}{\|q\|}\|_2 \,, \tag{2.1}$$

where $\hat{x}$ and $\hat{q}$ are ground truth position and orientation of an image and x and q are estimated position and orientation of the corresponding image computed by a forward pass of the network. The orientations q^* are denoted as quaternions (Appendix A.2) where $\hat{q}$ is a normalized quaternion. $\|\cdot\|$ denotes the norm respectively the L_2-norm. The loss function aims to minimize both, the position and the orientation term. However, for different tasks or applications one may want to minimize the position over the orientation or vice versa. Therefore, the weight or scaling parameter β is set manually and regulates this trade-off. To avoid the manual setting of a hyperparameter β, a loss function with respect to the re-projection error which combines position and orientation naturally is introduced in [KC17]. This weights the position and orientation depending on the scene and camera geometry. To define such a loss, a function π, which maps a 3D point $\mathbf{g}$ to 2D image coordinates $(u, v)^T$ is firstly considered as

$$\pi(\mathbf{x}, \mathbf{q}, \mathbf{g}) \rightarrow \begin{pmatrix} u \\ v \end{pmatrix} \,, \tag{2.2}$$

where x and q are the camera position and orientation. The function π is defined as

$$\begin{pmatrix} u' \\ v' \\ w' \end{pmatrix} = K(R\mathbf{g} + \mathbf{x}), \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u'/w' \\ v'/w' \end{pmatrix} , \tag{2.3}$$

where K is the camera's calibration matrix and R the mapping of $\mathbf{q}$ to the $SO(3)$ rotation matrix (see Appendix A for reference to $SO(3)$). The final loss is then defined as

$$\mathcal{L}_g(I) = \frac{1}{|\mathcal{G}'|} \sum_{\mathbf{g}_i \in \mathcal{G}'} \|\pi(\mathbf{x}, \mathbf{q}, \mathbf{g}_i) - \pi(\hat{\mathbf{x}}, \hat{\mathbf{q}}, \mathbf{g}_i)\|_1 , \tag{2.4}$$

where $\mathcal{G}'$ is the subset of all 3D points in the scene which are visible in the image I. The loss is therefore defined as the mean of all residuals computed from the points $\mathbf{g}_i \in \mathcal{G}'$.

This loss function boosts the performance of visual localization over the loss introduced in Equation 2.1 while avoiding a weighting parameter and no hyperparameter tuning by learning position and orientation simultaneously based on a reprojection error based loss. However, the reprojection loss could only be utilized for model refinement since it does not converge for random initialized models [KC17].

LSTMs [HS97] are a type of Recurrent Neural Network (RNN) [GK96] designed to accumulate relevant contextual information. RNNs are able to process temporal information by archiving an internal state to process sequences of input. Over iterative training steps, the internal states of previous training passes are taken into account during the present training pass, functioning as a memory. LSTMs have been applied with great success tackling handwriting recognition [GS09] and natural language processing [SVL14]. LSTMs are also successfully combined with CNNs considering pose regression [Wal+17]. The memory units reduce the dimensionality on the feature vector of networks like PoseNet, whereas the feature vector is treated like a sequence. A fully connected layer of 2048 neurons, representing a feature vector, as in the architecture of PoseNet, tends to overfitting, since the network is able to mostly learn the complete training data due to the high number of parameters – especially considering few training samples. LSTMs handle features from the convolutional and fully connected layers for better correlation by reducing the feature dimensionality. Four of such LSTM units are used to reduce the dimensionality in a structured way [Wal+17]. These four units are implemented to sequence over the fully connected layer, whereas the layer is reshaped to a 32×64 matrix. The memory units are then applied in up, down, left and right directions over this matrix. The four outputs are concatenated and serve as input for the fully connected pose layer. This dimensionality reduction prevents overfitting, likewise dropout layers, but is shown to work better as such. The memory units of the LSTM structure identify the most useful correlations for pose regression in the feature vector. The combination of a CNN architecture with LSTMs is shown to improve camera re-localization over PoseNet, which consists of a CNN only. The improvements are achieved on indoor and outdoor scenes of benchmark datasets.

3 Fundamentals of Deep Learning

In this chapter the fundamentals of deep learning are introduced with the general idea and theory of Artificial Neural Networks (ANNs) in Section 3.1 followed by Section 3.2 and 3.3 covering the fundamentals of CNNs and GANs.

3.1 Artificial Neural Networks

The idea of ANNs is firstly introduced by McCulloch and Pitts in 1943 [MP43]. They aimed to model the learning process of biological systems like the human brain. Ahead on this, Rosenblatt developed the first artificial neurons called perceptrons to model brain mechanics [Ros58]. ANNs consist of multiple such perceptrons or artificial neurons and are also called Multi-Layer Perceptrons. These networks aim to model biological neural processes but are not necessarily identical to them. A neuron in the brain is known to process an output y given an input x. This applies for artificial neurons as well and is given as

$$y = \sigma\Big(\sum_{i=0}^{N} w_i x_i + b\Big) = \sigma\big(\mathbf{w}^T \mathbf{x} + b\big) \ , \tag{3.1}$$

where x_i are the inputs with N as the number of input elements, w_i are the weights, b is the bias and $\sigma(\cdot)$ is an activation function. The weights and the bias are the learnable parameters that are adjusted during the training phase. The activation function is specified by the user. Activation functions are necessary to ensure that the model can achieve non-linearity and being able to generate non-linear models. Non-linear activation functions are therefore crucial for the system. The first proposed function to serve as an activation function is the step function for activating or deactivating neurons in a network. However this function leads to an unstable behavior throughout the training processes since small changes in the input may affect the output crucial. This unbalanced behavior handicaps the iterative adjustment of the learnable parameters and hinders the training process. Functions as *Sigmoid*, *Hyperbolic Tangent*, *ReLU* and *Leaky ReLU* suit the process of training better since small changes in the input do not necessarily impact the output too highly but rather ensure a more stable optimization. Figure 3.1 illustrates these common activation functions.

A perceptron and its biological inspiration are illustrated in Figure 3.2 and Figure 3.3. One such perceptron or artificial neuron could only learn a single linear function. By connecting multiple of these neurons forming an ANN, this limitation can be overcome.

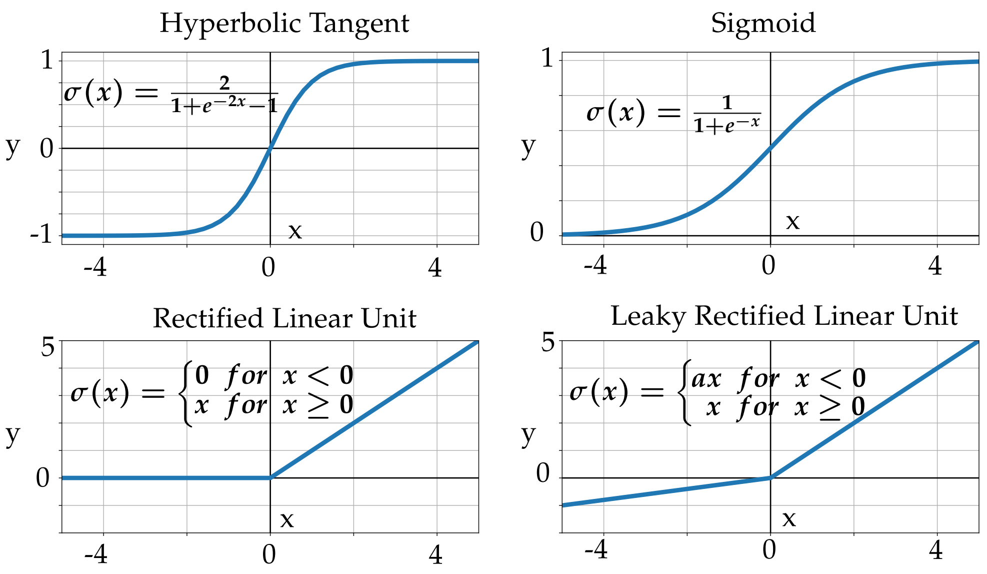


Figure 3.1: Activation functions to preserve non-linearity in neural networks.

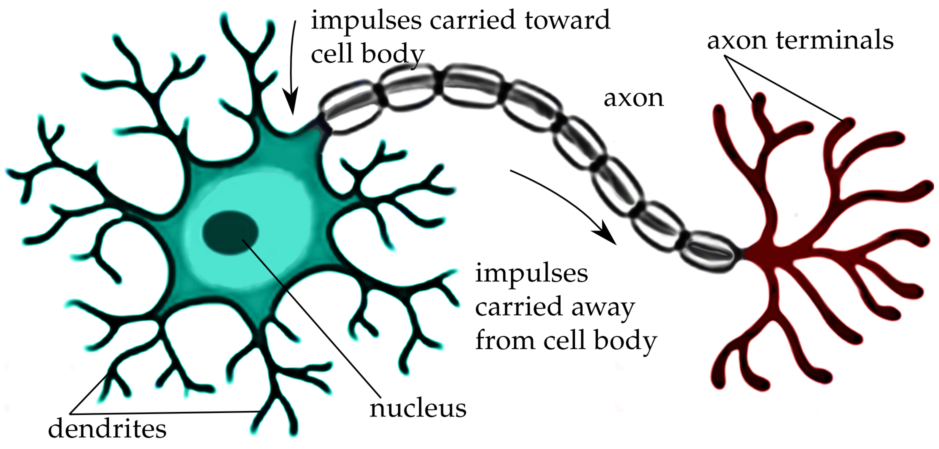

impulses carried toward cell body
axon terminals
axon
impulses carried away from cell body
nucleus
dendrites

Figure 3.2: Biological interpretation of a perceptron consisting of nucleus, axon, dendrites and axon terminals.

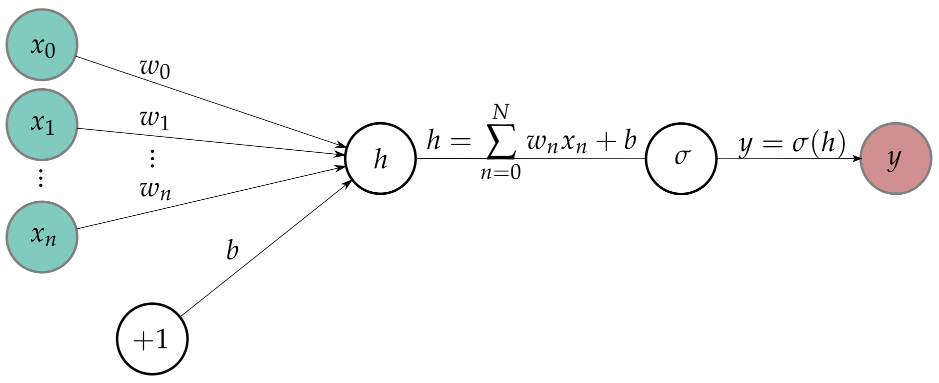


Figure 3.3: Artifical perceptron consisting of neurons, weights, bias, activation and output.

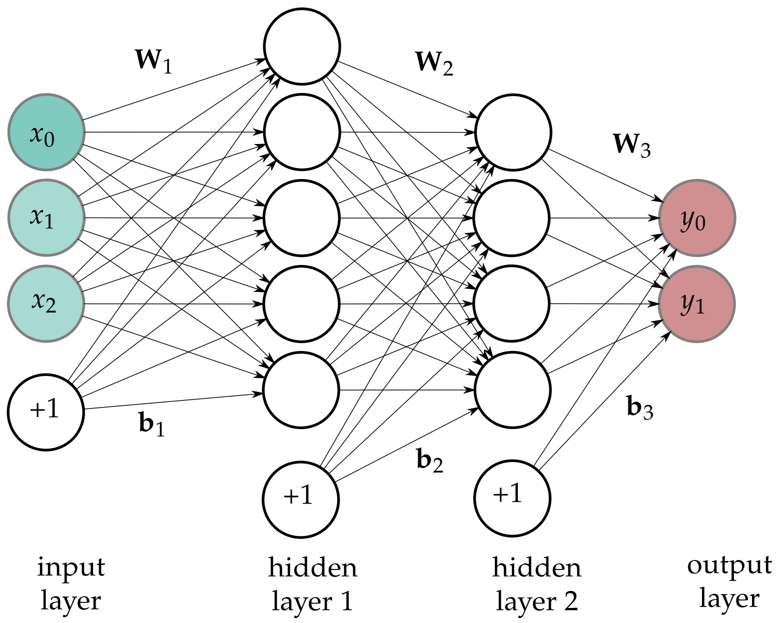


Figure 3.4: A neural network consisting of multiple perceptrons. This exemplary network has two hidden layers. The number of neurons of the input and output layer depends on the input and output data. The number of neurons of hidden layers depends on the general architecture.

ANNs consist of an input layer, an output layer and one or more hidden layers. Figure 3.4 illustrates such a network with two hidden layers. Networks with one hidden layer are called shallow neural networks, whereas networks with two or more hidden layers are called deep neural networks. All layers consist of neurons (depicted as circles in Figure 3.4) which are connected by their weights (depicted as arrows) to the neurons of the consecutive layer. Each layer has a bias unit marked with b in this figure. The number of neurons per layer can vary depending on the input data, the task and the general network architecture.

Denoting such a layer of multiple artificial neurons in matrix notation produces an output y as

$$y = \sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) = \sigma(h) \,, \tag{3.2}$$

where $\mathbf{W}$ is the matrix compounding all weights of one layer, $\mathbf{x}$ are the inputs to this layer, $\mathbf{b}$ is the vector of biases in this layer and σ is the activation function. Denoting the consecutive assembling of multiple artificial neurons in subsequent layers leads to

$$\mathbf{y} = \sigma_2(\mathbf{W}_2\, \sigma_1(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2) \,, \tag{3.3}$$

where $\mathbf{y}$ is the output vector of the network. This forward pass can be generally expressed for one layer as

$$\begin{aligned} \mathbf{h}_i^{in} &= \mathbf{W}_i\mathbf{h}_{i-1}^{out} + \mathbf{b}_i \\ \mathbf{h}_i^{out} &= \sigma_i(\mathbf{h}_i^{in}) \,, \end{aligned} \tag{3.4}$$

where $i = 1, ..., L-1$ with L being the total number of layers. For the first layer $\mathbf{h}_1^{in} = \mathbf{x}$ applies.

Neural networks with arbitrary numbers of hidden layers can be expressed with such chained matrix multiplications. This multiplication chain is known as forward pass computation and is applied to compute outputs given inputs. However, a crucial step that makes networks able to *learn* is the backward pass or backpropagation. A backpropagation algorithm developed by Werbos made training of multi-layer networks feasible and efficient [Wer74]. Neural network specific applications of efficient backpropagation are described by Werbos [Wer82]. By demonstrating the emergence of useful internal representation in hidden layers, backpropagation achieved high popularity for neural networks [RHW86]. In the backpropagation step the gradients of expressions are computed through recursive application of the chain rule. The gradient descent approach of backpropagation is an iterative and recursive method that updates the weights and biases of a network by optimizing a cost function or loss function. Such loss functions are a further crucial part of neural networks and will be handled later in this section. The entirety of weights $\mathbf{W}$ and biases $\mathbf{b}$ compose the set of learnable parameters of the network $\boldsymbol{\theta} = (\mathbf{W}, \mathbf{b})$. To perform backpropagation, the derivatives of all expressions within the network are required to be known. That includes calculating the derivative of the loss function with respect to the weights of the network. Backpropagation is explicit used by the gradient descent optimization algorithm to adjust the weights and biases by calculating the gradient of the loss function, whereas gradient descent uses the gradients for training the model by optimization. Therefore learning is considered an optimization problem.

During the training process, both the input and the output is provided in a supervised manner. The network processes forward propagations by simply computing the given inputs throughout the network by matrix multiplications. This is followed by the backpropagation step, updating the weights and biases in the network. These steps are iteratively performed whereas the weights are updated within every step, thus the network is optimized.

The loss function describes how the loss $\mathcal{L}$ is computed. This term is minimized during the training process. It is often expressed as a distance function or mean square error and can be expressed as

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = \mathcal{L}\Big(\hat{\mathbf{y}}^{(n)}, f_{\mathbf{W},\mathbf{b}}(\mathbf{x}^{(n)})\Big) = \|\hat{\mathbf{y}}^{(n)} - f_{\mathbf{W},\mathbf{b}}(\mathbf{x}^{(n)})\|_2\,, \tag{3.5}$$

with $f_{\mathbf{W},\mathbf{b}}(\mathbf{x}^{(n)}) = \mathbf{y}^{(n)}$ as the predictions of the last layer $\mathbf{h}_{L-1}^{out} = \sigma_{L-1}(\mathbf{h}_{L-1}^{in})$ computed by a forward pass (Equation 3.4) and $\widehat{y}$ as the labeled ground truth. The trainable parameters $\boldsymbol{\theta}$ of the network are initialized with small random values for the weights and zeros for the biases, by specific initialization schemes or with parameters achieved from previous training processes. Thus, initial parameterization can be realized by transferring weights from previous trainings by such transfer learning. Transfer learning supports faster convergence and helps to achieve higher accuracies [KGC15]. To enable transfer learning, the architecture of the source network has to fit the architecture of the target network. This can also be carried out partially for a network – e.g. for the first few layers – whereas the last layers of the network are initialized by random numbers. This is a popular scheme, since the first layers are known to learn low level features like edge and blob filters (similar to hand-crafted designed features) that are not too specifically related to the training data. The latter layers learn high level features, that are more dependent on the training data, thus are valuable to be re-trained.

After the initialization of the parameters, the loss is minimized by an optimization algorithm such as gradient descent. With gradient descent, the network's parameters are optimized in an iterative way, aiming to find the optimal values by minimizing the computed loss. The weights and biases are updated in an iterative way as

$$\begin{aligned} w_{i,j}^{(t)} &\leftarrow w_{i,j}^{(t)} - \lambda \frac{\partial E(\mathbf{W},\mathbf{b})}{\partial w_{i,j}^{(t)}} \\ b_{i}^{(t)} &\leftarrow b_{i}^{(t)} - \lambda \frac{\partial E(\mathbf{W},\mathbf{b})}{\partial b_{i}^{(t)}} , \end{aligned} \tag{3.6}$$

where i is the number of a neuron in layer t, j is the number of a neuron in layer $t+1$, $E(\mathbf{W},\mathbf{b})$ is the overall error given by the sum over all training examples in the training dataset $\mathcal{D}_{Train}$ and λ is the learning rate that determines the step size for parameter updates of $\boldsymbol{\theta}$.

The learning rate is usually set before training and is updated during the training process. Figure 3.5 and 3.6 illustrate a different choice of learning rates and their effect on the training process.

A learning rate set too low (blue) will lead to low convergence and the model is likely to get stuck in a local minimum. A learning rate set too high (red) will eventually fail to find the global minimum but will circulate around it or diverge. An optimal learning rate (green) will approach the global minimum fast with a high learning rate at the beginning and will come close to the global minimum by a decreased learning rate in the later training process.

The overall error $E(\mathbf{W},\mathbf{b})$ is given by the sum over all training examples in $\mathcal{D}_{Train}$

$$E(\mathbf{W},\mathbf{b}) = \sum_{n=1}^{N} \mathcal{L}\Big(\hat{\mathbf{y}}^{(n)}, f_{\mathbf{W},\mathbf{b}}(\mathbf{x}^{(n)})\Big) . \tag{3.7}$$

As stated above, backpropagation is applied for efficient training of the network. Therewith the partial derivatives from Equation 3.6 have to be computed. The derivatives can be computed as

$$\frac{\partial E(\mathbf{W},\mathbf{b})}{\partial w_{i,j}^{(t)}} = \frac{\partial \sum_{n=1}^{N_b} \mathcal{L}\Big(\mathbf{W},\mathbf{b},\mathbf{x}^{(n)},\mathbf{y}^{(n)}\Big)}{\partial w_{i,j}^{(t)}} = \sum_{n=1}^{N_b} \frac{\partial \mathcal{L}\Big(\mathbf{W},\mathbf{b},\mathbf{x}^{(n)},\mathbf{y}^{(n)}\Big)}{\partial w_{i,j}^{(t)}} , \tag{3.8}$$

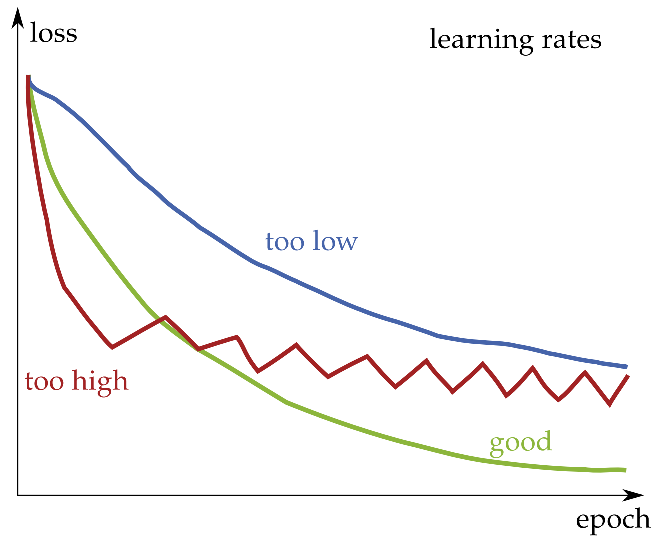


Figure 3.5: Different choice of learning rates and their influence on the loss over epochs. A learning rate set too low (blue) will lead to low convergence and the model is likely to get stuck in a local minimum. A learning rate set too high (red) will eventually fail to find the global minimum but will circulate around it or diverge. An optimal learning rate (green) will approach the global minimum fast with a high learning rate at the beginning and will come close to the global minimum by a decreased learning rate in the later training process.

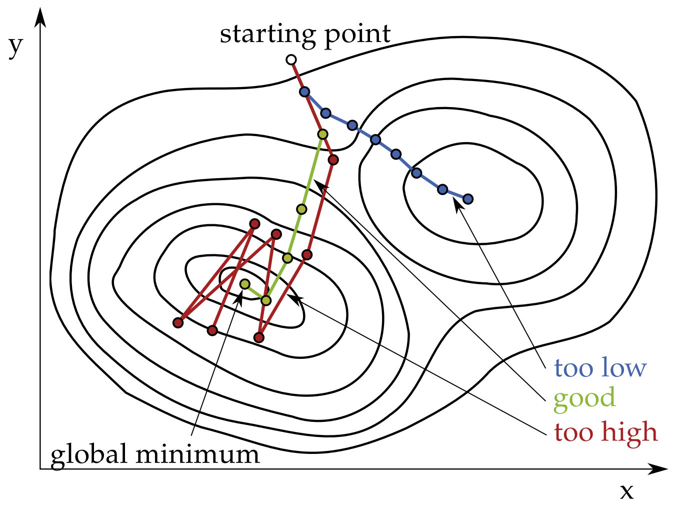


Figure 3.6: Different learning rates illustrated in a 2D scheme. The white circle represents the starting point as the initial parameter setting. A learning rate set too low (blue) is likely to find a local minimum but not the global minimum. A learning rate set too high (red) is not able to find the global minimum but will oscillate around it or diverge. An optimal or dynamic learning rate (green) decreases over time avoiding to get stuck in a local minimum while advancing preferable close to the correct global minimum.

where N_b denotes the number of input examples, also called batch size. The number of input examples is a subset of $\mathcal{D}_{Train}$. It is a common technique to split the dataset into small subsets of batches and evaluate the error over these subsets as a mean. This strategy speeds up the training process while providing faster convergence due to a more consistence data distribution.

The output of the network is computed by a forward pass (Equation 3.4). Therefore, the chain rule can be applied and the derivative of $\mathcal{L}(\mathbf{W}, \mathbf{b})$ for one training example $\left(\mathbf{x}^{(n)}, \mathbf{y}^{(n)}\right)$ can be denoted as

$$\begin{aligned}
\frac{\partial \mathcal{L}(\mathbf{W}, \mathbf{b})}{\partial w_{i,j}^{(t)}} &= \frac{\partial \mathcal{L}(\mathbf{W}, \mathbf{b})}{\partial h_i^{(t+1)}} \frac{\partial h_i^{(t+1)}}{\partial w_{i,j}^{(t)}} = \frac{\partial \mathcal{L}(\mathbf{W}, \mathbf{b})}{\partial h_{i-1}^{(t+1)}} \frac{\partial h_{i-1}^{(t+1)}}{\partial h_i^{(t+1)}} \frac{\partial h_i^{(t+1)}}{\partial w_{i,j}^{(t)}} \\
&= \left(\sum_{k=1}^{n_{t+2}} \frac{\partial \mathcal{L}(\mathbf{W}, \mathbf{b})}{\partial h_k^{(t+2)}} \frac{\partial h_k^{(t+2)}}{\partial h_{i-1}^{(t+1)}} \right) \frac{\partial h_{i-1}^{(t+1)}}{\partial h_i^{(t+1)}} \frac{\partial h_i^{(t+1)}}{\partial w_{i,j}^{(t)}} \\
&= \left(\sum_{k=1}^{n_{t+2}} \frac{\partial \mathcal{L}(\mathbf{W}, \mathbf{b})}{\partial h_k^{(t+2)}} w_{k,i}^{(t+1)} \right) \sigma'\left(h_i^{(t+1)}\right) h_{j-1}^{(t)} ,
\end{aligned} \tag{3.9}$$

where n_{t+2} is the number of units in layer $t+2$. The error δ_i^t that is propagated backwards through the network and can be defined by

$$\delta_i^t := \frac{\partial \mathcal{L}(\mathbf{W}, \mathbf{b})}{\partial h_i^{(t)}} = \left(\sum_{k=1}^{n_{t+1}} \sigma_k^{(t+1)} w_{k,i}^{(t+1)} \right) \sigma'\left(h_i^{(t)}\right) \tag{3.10}$$

for layers with $i = 1, ..., L-1$ and

$$\delta_i^L := \frac{\partial \mathcal{L}(\mathbf{W}, \mathbf{b})}{\partial h_i^{(L)}} = \frac{\partial \mathcal{L}(\mathbf{W}, \mathbf{b})}{\partial a_i^{(L)}} \frac{\partial a_i^{(L)}}{\partial h_i^{(L)}} = \frac{\partial \mathcal{L}(\mathbf{W}, \mathbf{b})}{\partial a_i^{(L)}} \sigma'\left(h_i^{(L)}\right) \tag{3.11}$$

for the output layer with $i = L$. The error terms δ_i^t store the errors of each neuron i in layer t and is propagated backwards through the network starting with δ_i^L.

The algorithm of training a neural network can thus be depicted as follows:

1. Initialization of the network parameters $\boldsymbol{\theta} = \mathbf{W}, \mathbf{b}$.
2. Computation of the forward propagation of $\mathbf{h}_i^{out}$ for every layer (Equation 3.4).
3. Computation of the error terms δ_i^t (Equations 3.10 and 3.11).
4. Update of the parameters according to Equation 3.6 and repeat from step 2.

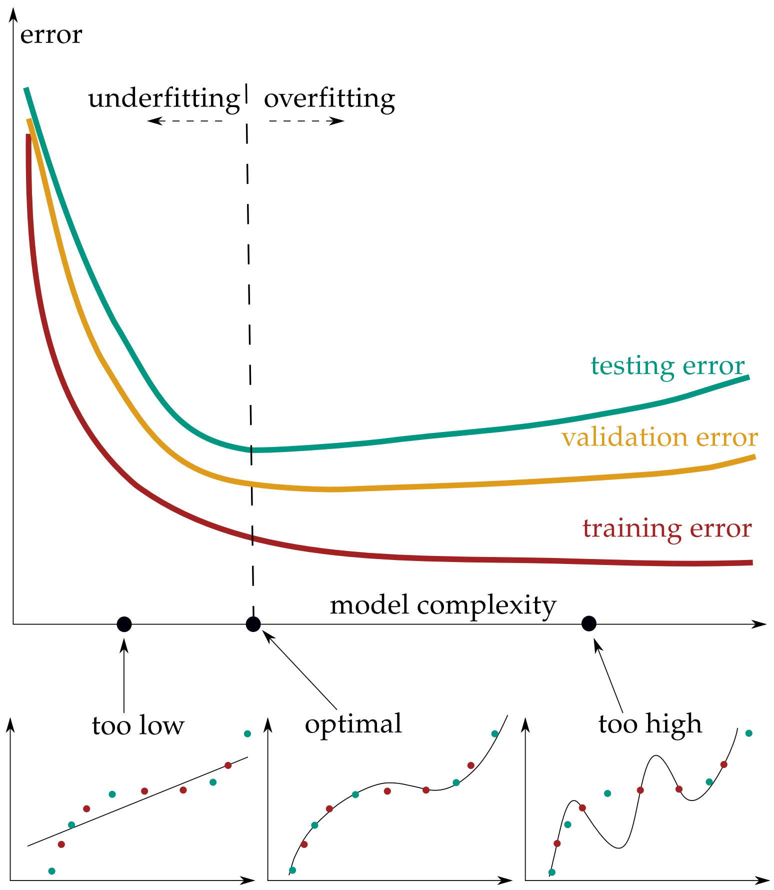


Figure 3.7: Relationship of how the model complexity leads to underfitting and overfitting and thus affects the testing error.

An important prerequisite for a reasonable training procedure is the splitting of training data. Generally the dataset, which a model is trained on, is called training dataset. To supervise the training procedure a validation dataset should be utilized in addition. This dataset serves for frequently validating the trained model and to intervene by adjusting hyperparameters or interrupt the training process in an early stage if necessary. In contrast to the training dataset, the validation dataset does not contribute to the adjustment of the learnable model parameters $\boldsymbol{\theta}$ directly. The network's input is solely data from the training dataset. With progressing training, the model is fitted more and more to the training data. The validation set helps to find the breaking point between underfitting and overfitting of the model. The relationship of how model complexity leads to underfitting and overfitting and affects the testing error is depicted in Figure 3.7.

3.2 Convolutional Neural Networks

CNNs are a kind of ANNs for processing data that has a known grid-like topology where a special kind of layer is utilized that shares parameters between units by so-called convolutions. Therefore, this type of layer is called convolutional layer. A pioneering paper on CNNs is published applying such networks for hand-written character and digit recognition being applied practically to read several million checks per day [LeC+98]. Generally convolutional networks are neural networks that use convolution operations in place of a general matrix multiplication in at least one layer of the model. The convolution operations, also called kernels, takes a multi-dimensional quadratic grid of neurons in layer t and a set of s filters as input. Each kernel $k^{(t,s)}$ has a size of $(2r^{(t)} + 1) \times (2r^{(t)} + 1)$. These kernels contain the learnable weights of the network. Depending on the size of a kernel, only a part in layer t is connected to a single neuron in layer $t + 1$. In contrast to fully connected layers, convolutional layers are sparse connected layers. The rectangular region in the input layer t that is connected to a neuron in layer $t + 1$ is called the receptive field. The size of the receptive field depends on the size of the convolution filter and increases with network depth considering the original input. The neurons which are activated by the convolution determine the output volume of layer $t + 1$, which is called activation map or feature map (Figure 3.8). A convolutional layer is defined by the hyperparameters

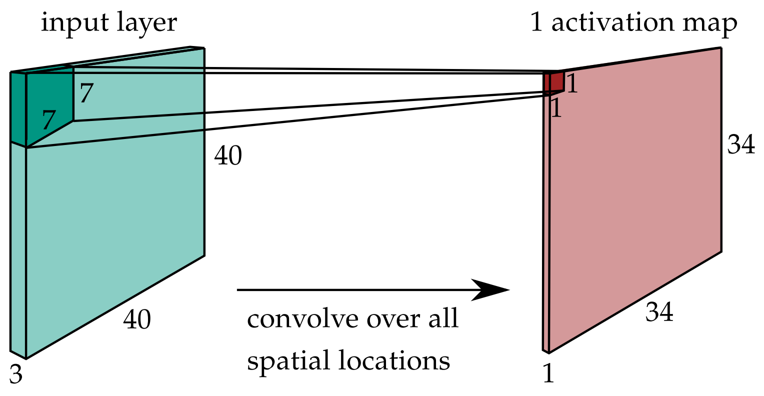


Figure 3.8: Convolutional layer. In this exemplary illustration the input layer has a dimension of 40×40 and a 7×7 filter is applied for convolutions. The generated activation map has a dimension of 34×34 when no zero-padding is applied and the convolution is carried out with a stride of 1. The input layer has 3 channels (e.g. a RGB image patch), whereas the filters third dimension has to be 3 accordingly. The activation map is reduced to one channel after the convolution. The third dimension of the final output volume equals the number of convolution filters that are applied to the input. Only the first convolution is depicted exemplary in this figure.

filter size, stride and zero padding. The filter size is simply the width and height of the convolution filter, that is the quadratic field of neurons that serve as input from layer t. The stride determines the step size, which the filter is shifted for each convolution. The stride is usually set to a small number as 1 or 2 to not decrease the resolution of the subsequent layers throughout the network too much. The zero padding is a technique to preserve the layer size of t after a convolution. Filter sizes larger than 1×1 and a stride larger than 1 reduce the size of the layers. To avoid such behavior, zeros are added to the border of an input layer. The number of filters of a convolutional layer determine the depth of the output layer. The activation maps are stacked and form a 3D layer. The depth of the convolutional filters of layer $t + 1$ equals the number filters in the respective precedent layer t. The depth of the initial input layer represents the number of spectral channels of the input image. Figure 3.8 illustrates a convolutional layer with its receptive field, activation map and hyperparameters.

Convolutional layers utilize a set of weights (filters) to generate an activation map, whereas fully connected layers utilize single weights for each pair of neurons in layers t and $t+1$. Therefore the number of learnable parameters for a convolutional layer is generally multiple-times smaller than the number of weights for a fully connected layer. That is an efficient regularization technique and helps to speed up the training process. This parameter sharing also supports better generalization of the network since particular features occur potentially at several locations within an image and can be extracted with the same filter.

The number of weights can be further decreased by the utilization of pooling layers in between convolutional layers. A pooling layer decreases the dimension of the activation map by transferring the maximum or the average value of a limited region from the previous layer (Figure 3.9). This reduces the number of learnable parameters considering fully connected layers and aims to alleviate overfitting by limiting the model complexity. The relation between the error from overfitting and underfitting is depicted with respect to the model complexity in Figure 3.7. However, in the pooling process information is irrevocable lost. An alternative method is to increase the stride of a convolutional layer which also decreases the size of the subsequent layer, but without a general loss of information. A further technique that helps prevent overfitting is dropout. Dropout randomly deactivates neurons in the network during particular training iterations. Therefore, a slightly different network is trained at each iteration, driving the network to learn a more general representation.

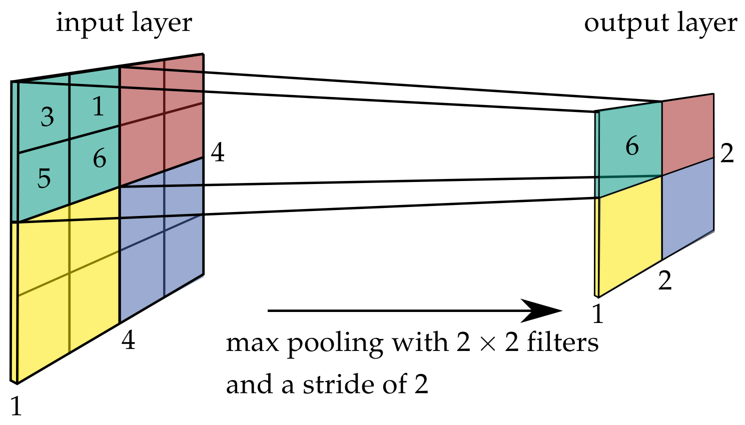


Figure 3.9: Pooling layer. In this exemplary illustration the input layer has a dimension of 4×4 and a 2×2 filter is applied for pooling. The generated output layer has a comprised dimension of 2×2 when no zero-padding is applied and the pooling is carried out with a stride of 2. The overall size of the input layer is reduced by a factor of four by the pooling layer. A loss of information is a drawback of such pooling layers. Alternatively, an average pooling layer would transfer more information to the next layer. Only the first pooling operation is depicted exemplary in this figure.

3.3 Generative Adversarial Networks

GANs are a machine learning architecture consisting of two networks which compete with each other, whereas one network generates artificial data and the other rates this data as real or fake. Introduced by Goodfellow et al. [Goo+14] GANs are successfully established in several fields of computer vision including image inpainting [Yua+19], semantic segmentation [Luc+16], image-to-image translation [Ano+19; Ano+18] by CycleGAN [Zhu+17] or text-to-image synthesis [Ree+16].

The two competing networks are a generator and a discriminator. The generator network G produces fake images whereas the discriminator network D tries to distinguish the fake images produced by the generator from real images given by a training dataset. Both networks are jointly trained in a competitive way. The goal is to train the generator network to map random noise $\mathbf{z}$ to output images $\mathbf{y}$ as artificial image samples. In other words, the goal of G is to estimate the data distribution of a training dataset as good as possible and to generate samples from this learned distribution. To train G, the discriminator network D is introduced to optimize G by distinguishing between real images $\mathbf{y}$ and fake images $\mathbf{y} = G(\mathbf{z})$ generated by G. Formally, the generator maps a noise vector $\mathbf{z}$ in the latent space to an image $\mathbf{y}$

$$G(\mathbf{z}) \rightarrow \mathbf{y}\,, \tag{3.12}$$

whereas the discriminator is defined as

$$D(\mathbf{y}) \rightarrow [0, 1] \tag{3.13}$$

and classifies an image $\mathbf{y}$ as real (close to 1) or as fake (close to 0). The two networks are trained in a competitive fashion with backpropagation. The loss function is generally formulated as

$$\min_G \max_D \mathcal{L}_{GAN}(G, D) = \min_G \max_D \mathbb{E}_{\mathbf{y}\in\mathcal{Y}}[logD(\mathbf{y})] + \mathbb{E}_{\mathbf{z}\in\mathcal{Z}}[log(1 - D(G(\mathbf{z})))]\,, \tag{3.14}$$

where $\mathbb{E}$ denotes the expected value, $\mathcal{Y}$ the set of real images and $\mathcal{Z}$ denotes the latent space. This loss function (Equation 3.14) is called adversarial loss. The term $\mathbb{E}_{\mathbf{y}\in\mathcal{Y}}[logD(\mathbf{y})]$ represents the predicted log probability of D that $\mathbf{y}$ is real and the term $\mathbb{E}_{\mathbf{z}\in\mathcal{Z}}[log(1 - D(G(\mathbf{z})))]$ represents the predicted log probability of D that $G(\mathbf{z})$ is fake. D is generally a classification network that determines the probability that an image belongs to class 0 (fake) or 1 (real). After completion of the training procedure the generator network G is used to generate artificial data, while D is only utilized during the training process to optimize the former.

Specifically, the networks are optimized by alternating the training of D and G by maximizing the GAN loss with respect to the parameters of the discriminator network $\boldsymbol{\theta}_D$ and then minimizing that loss with respect to the parameters of the generator network $\boldsymbol{\theta}_G$. Therefore, D tries to get the term $D(G(\mathbf{z}))$ from Equation 3.14 close to 0. That is, when all (fake) images generated by G are detected and labeled accordingly correct as fake by D. On the other hand, G tries to get the term $D(G(\mathbf{z}))$ close to 1. That is, when all (fake) images are not detected by D and wrongly labeled as real. With this strategy G will consequently learn an estimation of the real data distribution from the training dataset. The discriminator network is trained from fake samples, generated by G and real samples given from the training set. The general scheme of a GAN is illustrated in Figure 3.10.

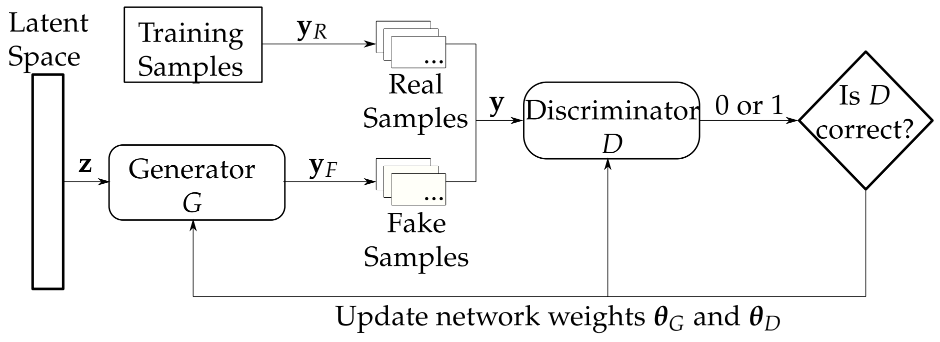


Figure 3.10: Overview of a GAN scheme. The generator G produces images $\mathbf{y}_F$ as fake samples given a noise vector z from a latent space. The discriminator D takes real images $\mathbf{y}_R$ from the training samples and fake samples as input and classifies them as fake (0) or real (1). Depending on the outcome, the network weights of the generator network $\boldsymbol{\theta}_G$ and discriminator network $\boldsymbol{\theta}_D$ are updated. When the training converges, the generator improves and generates fake images that look like images from the training dataset.

In practice, the first image samples produced by G are of low quality. The discriminator D will therefore quickly learn this discrepancy and easily detect the fake images while labeling them correctly with 0. That means that $D(G)(\mathbf{z})$ will be close to zero. That leads to $log((1 - D(G(\mathbf{z})))$ being close to zero and hence the loss of the generator and its gradients are close to zero as well. Small gradients for G will slow down the training process of the generator network extremely. It is therefore a practical way to to minimize $-log(D(\mathbf{y}, G(\mathbf{z})))$ instead of $log((1 - D(G(\mathbf{z})))$. This modification leads to stronger gradients for G and keeps the optimum unchanged.

The overall training problem of a GAN is a minimax game. The optimal solution of such a problem is referred to as Nash Equilibrium [Nas51] in game theory. The Nash Equilibrium for GANs can be defined as the local maximum of $\mathcal{L}_{GAN}$ with respect to $\boldsymbol{\theta}_D$ and a local minimum of $\mathcal{L}_{GAN}$ with respect to $\boldsymbol{\theta}_G$. At this point – which could be pictured as a saddle point – G and D are best optimized. Compared to the training of a neural network, finding a saddle point is more difficult than finding a single minimum or maximum. However, GANs have the advantage to not tend to overfitting, since the generator network is getting information about the training data only in an indirect way via the discriminator, therewith it is unable to learn a direct mapping by simply replicating the training data.

Images generated with GANs are usually less blurred and more realistic compared to other generative models such as Variational Auto Encoders [KW13]. Optimal GANs, where the generator produces perfect images and the discriminator always produces 0.5 are theoretically proven to exist [Goo+14].

4 Datasets

There are several benchmark datasets for image re-localization. The *7 Scenes* dataset is popular for evaluating indoor applications [Sho+13b]. The *Cambridge Landmarks* benchmark is an outdoor dataset consisting of six outdoor scenes [KGC15]. Camera re-localization datasets generally consist of images (separated in training and evaluation datasets) and their corresponding poses. Optionally, datasets provide depth images, 3D models, or images in varying domains such as day and night, summer and winter or others. In the following sections, the most important datasets acquired and/or utilized in this thesis are presented. The *Atrium*, *Puzzle* and *Shop Façade* datasets are introduced which are used for training and evaluation in the experiments throughout this thesis.

4.1 Atrium

The *Atrium* dataset is an outdoor set consisting of captured images showing the atrium of a building. The dimension of the scene is approximately $39 \times 36 \times 18\ m^3$. The training dataset consists of 864 captured images collected within the *LaFiDa*[1] benchmark collection [UJ17]. The images are captured by two different DSLR cameras and a smartphone camera.

Furthermore, the dataset contains two evaluation datasets, the *medium coverage* set and the *low coverage* set. The images for these sets are captured with a $4K$ resolution camera attached to an UAV. The *medium coverage* set consists of 145 images captured at ground level which show a medium coverage to the training data. The *low coverage* set consists of 198 images captured at an altitude of up to $18m$ which are spatially far away and have high discrepancies in perspectives compared to the images of the training dataset. As no images with high altitude and a downward facing field of view are present in the training data, this evaluation set shows a low coverage to the training data and represents challenging characteristics.

A high coverage can be stated, if training and evaluation images share similar poses, e.g. when they are spatially close to each other (position) and share a similar field of view (orientation). The *low coverage* set is therefore challenging for a pose regression CNN, as the network is not trained on similar images nor poses during the training process with respect to the evaluation data. The image poses for the training and evaluation images are derived by SfM [Agi17]. Side views of the 3D model are shown in Figure 4.1. Example images of the *Atrium* dataset are shown in Figure 4.2. Figure 4.2 (a) shows an evaluation

[1] https://www.ipf.kit.edu/lafida.php (last access 10th September 2019)

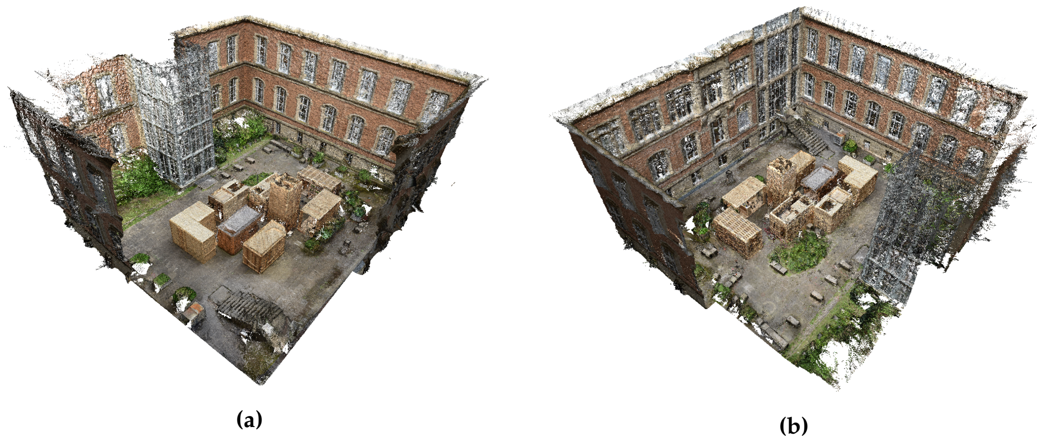


Figure 4.1: Side views of the reconstructed 3D model of the atrium. The red bricked walls show very similar structures, leading to a challenging scene for computer vision algorithms.

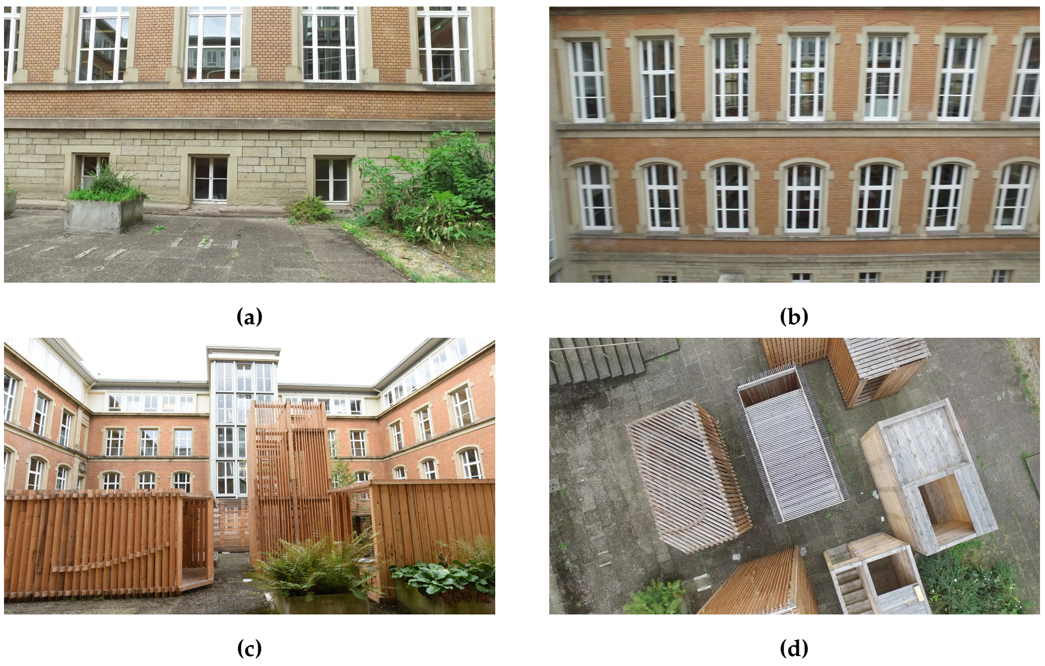


Figure 4.2: Image examples of the atrium. (a) Windows in the scene cause reflections, this is challenging for computer vision tasks. (b) Image captured with high angular motion and therefore shows motion blur. (c) Image shows ambiguous structures. (d) Evaluation image of the *low coverage* set captured at high altitude. A similar perspective is not present in the training dataset, therefore pose is unsimilar to any training pose.

image depicting reflections in the upper windows. Figure 4.2 (b) shows image blur caused by angular motion during the image acquisition process, both effects are challenging for general computer vision tasks. Figure 4.2 (c) shows the ambiguous structures of the walls. Figure 4.2 (d) shows an evaluation image of the *low coverage* set. A similar image is not contained in the training dataset. The position as well as the orientation are far away from any training pose.

4.2 Puzzle

The *Puzzle* dataset is a small sized set of captured images of a usual jigsaw puzzle showing an ancient map of a world with some embellishments. The spatial dimensions of the puzzle are approximately $114 \times 83\ cm^2$. The base *Puzzle* dataset consists of 19 images captured in nadir perspective of a puzzle serving as training images and additionally 48 captured images serving as evaluation images. The images are captured with a smartphone camera. A photo-realistic 3D model is generated from these 19 captured images. Figure 4.3 shows the model in nadir view. The images and the 3D model are utilized in Chapter 6.

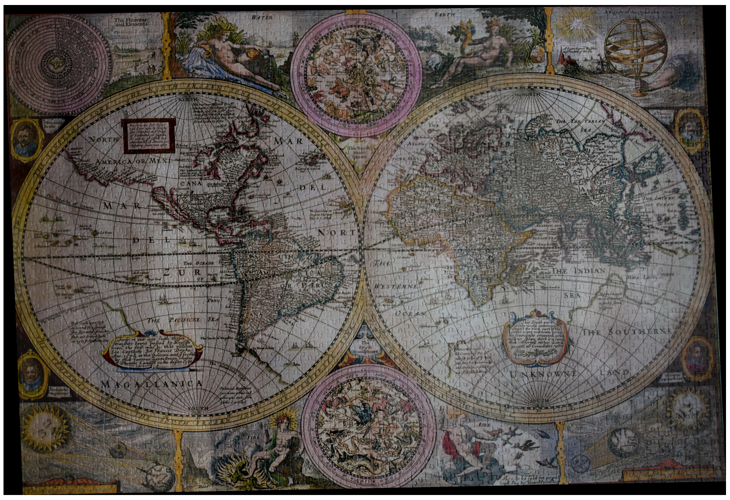
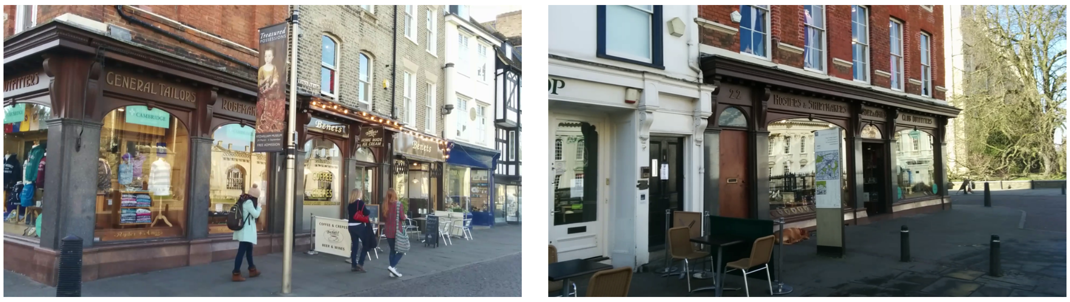


Figure 4.3: Nadir view of the reconstructed 3D model from the *Puzzle* dataset. This model is generated with 19 captured images and serves for DA.

4.3 Shop Façade

The *Shop Façade* dataset is an outdoor set of captured images mostly showing two façades of a shop. The dimension of the area in which the images are captured is denoted with $35 \times 25\ m^2$. The dataset generally serves as a benchmark for camera re-localization and contains three video sequences recorded with a smartphone camera and their extracted image frames separated in 231 training and 103 evaluation images. A pose is provided for every image. The dataset is part of the Cambridge Landmarks benchmark [KGC15] and is available online[2]. Two example images are illustrated in Figure 4.4. High coverage between training and evaluation data can be stated for this dataset. A 3D model, shown in Figure 4.5, is generated from the training images and serves for image-to-image translation in Chapter 7.

[2] http://mi.eng.cam.ac.uk/ (last access 4th October 2019)



(a) **(b)**

Figure 4.4: Example images of the *Shop Façade* dataset. The dataset images show two façades of a shop, including reflections and dynamic objects in the scene.

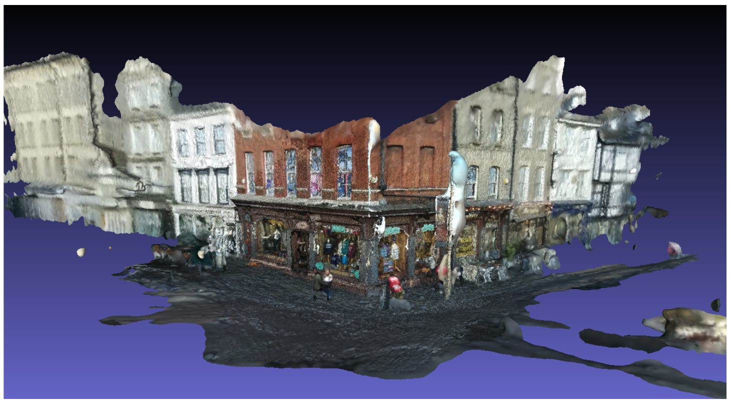


Figure 4.5: Front view of the reconstructed 3D model of the *Shop Façade* dataset. This model is generated from the training dataset and serves for image rendering in Chapter 6. The illustrated view of the model is generated with *meshlab* [Cig+08].

5 Localization without Data Augmentation

This chapter is focused on visual localization without DA, whereas the main interest lies in the benefits of lightweight CNNs for the localization of mobile devices such as UAVs. The general performance of such data-driven end-to-end approaches for localization is investigated, whereas prioritization is set on the general model size of utilized networks. Fields that potentially profit from camera localization is autonomous driving, pedestrian navigation or AR and VR. The devices on which such localization pipelines are embedded are computationally weak due to space limitations of mobile platforms. A network architecture with a significantly reduced number of parameters compared to typical recently presented architectures is introduced in this chapter. Investigative experiments are carried out in Chapter 8.

Parts of this chapter have been published in the conference paper

> M. S. Müller, S. Urban and B. Jutzi. SqueezePoseNet: Image Based Pose Regression with Small Convolutional Neural Networks for Real Time UAS Navigation. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences - International Conference on Unmanned Aerial Vehicles in Geomatics*. Vol.: IV-2/W3, 49–57. 2017.

The chapter is outlined as follows. A brief introduction is given in Section 5.1 focusing on localization methods by CNNs and their model sizes. Related work is reviewed in Section 5.2 covering different kinds of localization solutions as well as CNN-based approaches. A modified network for localization with a lightweight architecture is introduced and outlined in Section 5.3. Subsequently the training procedure is clarified in Section 5.4. All experiments are described in Chapter 8.1, whereas the utilized data for the experiments is presented in Chapter 4.1.

5.1 Introduction

Camera localization is a valuable part of navigation systems for robots, aerial vehicles, cars or pedestrians and has high potential to improve such. The number of Unmanned Aerial Systems (UASs) operating in commercial and private fields increased with the rise of robust and easy-to-use UAVs. Hence, the research in the fields of navigation, path planning, localization, data acquisition or obstacle avoidance for such vehicles is increasing. These systems need reliable navigation solutions which are mostly based on GNSS and are likely combined with alternative methods such as Inertial Navigation Systems. As failures due to gaps in signal coverage caused by occlusions or multi-path effects weaken the navigation solution via satellites, alternative methods for a reliable navigation of UASs are desired. GNSS provides a position estimate in a global reference system. Alternative methods for navigating in a global reference system, are usually based on digital surface models, digital terrain models or CAD models. The large memory consumption of such models renders these approaches unfeasible for on-board processing on computational weak devices which are mounted on UAVs, small robots or used by pedestrians.

Further methods to obtain a global position are aerial image matching using feature detection, correlations or Neural Networks. Feature detection or correlation-based methods are computational intensive, especially for navigating in a huge area where potentially millions of descriptors have to be stored and matched. Thus, losing real time capability and making them inefficient for real-time navigation. CNNs in contrast perform forward passes even on small on-board Graphical Processing Units in little time while having a limited memory demand and power consumption [Nvi17]. Considering small platforms, nano drones demand processing devices that are lightweight and consume low power [Pal+19]. CNNs with small model sizes fit the conditions of such devices better than CNNs with large model sizes.

Local navigation methods like visual SLAM or VO on the other hand provide potential solutions for relative positioning [ESC14; MMT15; EKC17]. However, these methods lack navigation in a global reference system without providing additional information, like relevant geo-referenced key points or the initialization to a referenced model. For short travel-paths these methods have a high accuracy but are subject to drift when the traveled distance increases. However, considering camera re-localization, SLAM and VO are a dissimilar tasks. Camera re-localization is a global localization task, that focuses on estimating an image pose in a global reference frame, whereas SLAM and VO approaches estimate relative poses between consecutive image frames and are therefore considered local localization approaches. Global localization methods are operational as standalone systems, whereas approaches with a high frame rate provide a dense trajectory like local localization methods. Running complex localization on computational weak computers or embedded devices may lead to limited real-time capability and inadequately results. Considering running a localization framework on small UAVs, respectively Micro Aerial Vehicles (MAVs), the computation power is limited due to maximum payload of such vehicles and the along going fact that small and light processing devices have lower computational power than common processors. Localization approaches that aim to be deployed for mobile localization are therefore subject to the limited computational power of utilized on-board processing units and need to be carefully designed with the target hardware in mind.

A solution for this problem could be provided by CNNs. CNNs have little processing requirements during runtime, since only forwards passes have to be computed contrary to the training of CNNs which is computational ambitious, due to the computational expensive backpropagation. The training can be performed offline and is not impacting the feasibility of CNN-based localization approaches considering mobile platforms. Considering deep CNNs – like the VGG16-Net [SZ14b] – with a high number of parameters, the storage (some hundreds of megabytes) and processing requirements tend to exceed the capability of on-board devices and are cumbersome to process on weak processing devices. A forward pass takes up to a few seconds loosing real-time requirements for real-world applications. In contrast to this, a small CNN which is efficient in terms of processing while maintaining a satisfying accuracy for localization is desirable. For this purpose, a CNN-based solution for the localization is introduced in this chapter. A variant of SqueezeNet [Ian+16], which is a CNN that solves classification while maintaining accuracy of deeper nets like AlexNet [KSH12] built with 50$\times$ less parameters. This network is modified to solve for pose regression. The modified network is called SqueezePoseNet, since it is adapted from SqueezeNet but solves for poses.

5.2 Related Work

In general, localization can be tackled in various ways. Concerning GNSS-free localization, image-based approaches are able to determine poses in a global reference frame. Hand-crafted solutions are provided by finding correlations between aerial images and images taken by UAVs for subsequent image matching followed by the localization of the aerial vehicle [CD09; CD11]. Methods based on feature matching utilize remotely sensed data [Li+09] or oblique images [HSY12] to find image poses. 3D models or building models are used to determine camera poses from images for AR [RD06a] or the estimation of UAV positions [URH16]. For real-time localization in indoor scenes, simple CAD models are utilized to get spatial information [UWS09; Urb+13; MV16]. CNNs are utilized to find matching pairs of aerial and UAV images [Alt+16] or matches between terrestrial and UAV images [Lin+15].

Alternatively to these methods that provide a localization solution with respect to a global reference frame, solutions that provide poses in a local frame are popular for relative localization. These approaches include visual SLAM or VO, which both reconstruct a trajectory of image poses by determining relative poses between consecutive image frames. Efficient solutions are ORB-SLAM [MMT15], Large-Scale Direct Monocular SLAM (LSD-SLAM) [ESC14] or Direct Sparse Odometry (DSO) [EKC17]. These methods provide satisfying solutions according to accuracy and real-time capability. However, providing localization in a global reference frame can only be achieved by fusing these local solutions with global localization methods. Even though SLAM or VO solutions show impressive results and do drift only slightly for short trajectories, they will drift over long distances particularly if there are no loop closures or when no global information is provided. The restriction to global information leads to failures of such approaches, especially when the track is lost. In that case, restoring a track is impossible without moving back to a known or mapped position.

Camera re-localization in a data-driven end-to-end pipeline by CNNs is firstly tackled successfully with PoseNet [KGC15], a CNN that determines image poses of a known scene. For this purpose the CNN is trained with images and their corresponding poses in order to estimate the pose of an unseen image. Hence the training is carried out with global poses, the pose estimates during runtime are determined with respect to this reference frame. In contrast to SLAM or VO, every particular pose estimate is independent from previous or posterior pose estimates. An enhancement of this approach is the Bayesian PoseNet [KC16] which provides an additional localization uncertainty by adding dropout layers after each convolutional layer. Furthermore the localization accuracy is improved by simply averaging over multiple forward passes. For outdoor datasets the accuracy for a pose estimate is approximately $2\ m\ and\ 6^{\circ}$. The architecture of PoseNet is based on the architecture of GoogLeNet [Sze+15], a CNN to solve classification tasks like the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [Rus+15; Den+09]. Enhanced accuracies for data-driven learning are derived by further improvements using LSTM layers, with a type of RNN which is combined with CNNs [Wal+17]. LSTM handles the problem of a dissolving gradient during the backpropagation using so called gates. LSTM not only shows great success in handwriting or speech recognition but also in localization.

The main target of this chapter is to investigate data-driven learning approaches for localization with focus on the networks model sizes. Considering, that deep CNNs with a large number of parameters need a higher computation power than smaller CNNs, the latter is preferred. Exemplary, the VGG16-Net [SZ14b] has a model size of 528 MB. The model size of PoseNet or the LSTM-based CNN which are both based on GoogLeNet have sizes of about 50 MB. In Section 5.3, SqueezeNet, a small CNN designed for classification tasks is adapted and modified for camera re-localization. SqueezeNet is especially designed with a reduced number of weight parameters to keep the model size low. With a model size of only 4.8 MB, this network is 10 times smaller than GoogLeNet.

5.3 Methodology

For the demands of camera re-localization, the architecture of SqueezeNet is adapted and modified to solve for pose regression. Therefore the architecture that is designed to solve classification tasks with a distinction of 1000 classes is modified. The original design strategy with little parameters lies in the fire modules. Each fire module first decreases the number of input channels from the previous layer by 1×1 convolutions in a squeeze operation [Ian+16]. Subsequently an expand operation that is a combination of 1×1 and 3×3 filters increases the number of activation maps while keeping the number of parameters low. The architecture of a fire module is depicted in Figure 5.1.

In addition, the SqueezeNet architecture lacks a final fully connected layer as these layers increase the number of parameters significantly. It is instead substituted by a final convolutional layer, that consists of as many 1×1 filters as classes. Subsequently average pooling is used to yield a vector whose length equals the number of target classes. To modify for pose regression, the classification layer is substituted by a fully connected layer

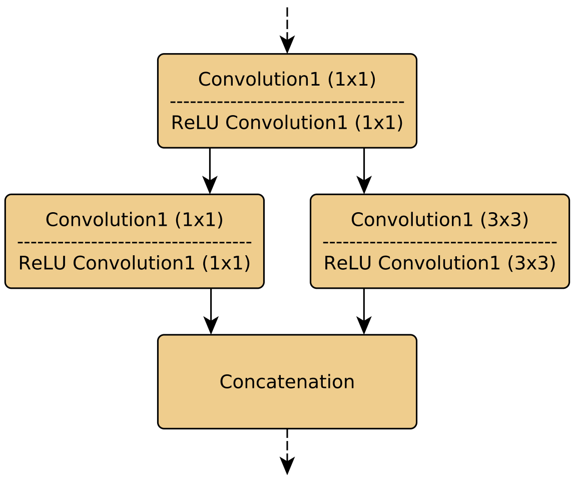


Figure 5.1: Architecture of a fire module. A squeeze operation [Ian+16] is performed by the 1×1 convolutional layer. Subsequently an expand operation that is a combination of 1×1 and 3×3 filters increases the number of activation maps while keeping the number of parameters low.

with 500 neurons. This layer serves as a descriptor and is meant to enable the distinction of different poses. Finally, two fully connected layers for actual pose estimation are added. A three neuron layer is added for the position component and a four neuron layer is added for the orientation component. The rotation is parametrized as quaternions for all computations.

Figure 5.2 depicts the architecture of the modified network. It is named SqueezePoseNet since it solves camera re-localization by pose regression, while holding the squeeze operations from the fire modules.

Further modifications include, the substitution of the activation functions from ReLU to Leaky ReLU [MHN13]. This is meant to help convergence. In addition, batch normalization [IS15] is added after each convolutional layer, making higher learning rates possible.

For evaluation purpose, an additional CNN, the VGG16-Net [SZ14b], which is designed to solve classification tasks is modified in a similar manner as SqueezeNet to solve for pose regression. Two dense layers for pose regression are added and the layers activation functions are set to Leaky ReLUs. This modified network is deeper and has more parameters than SqueezePoseNet, which leads to a bigger model size of more than 500 *MB*. Deeper networks generally tend to be more accurate than small ones [Ian+16]. CNNs are optimized during the training process by iteratively updating their weighting parameters using backpropagation. To optimize SqueezePoseNet for pose regression the following loss function is minimized [KGC15]

$$Loss_i = \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2 + \beta \, \|\mathbf{q}_i - \frac{\hat{\mathbf{q}}_i}{\|\hat{\mathbf{q}}_i\|}\|_2 \,. \tag{5.1}$$

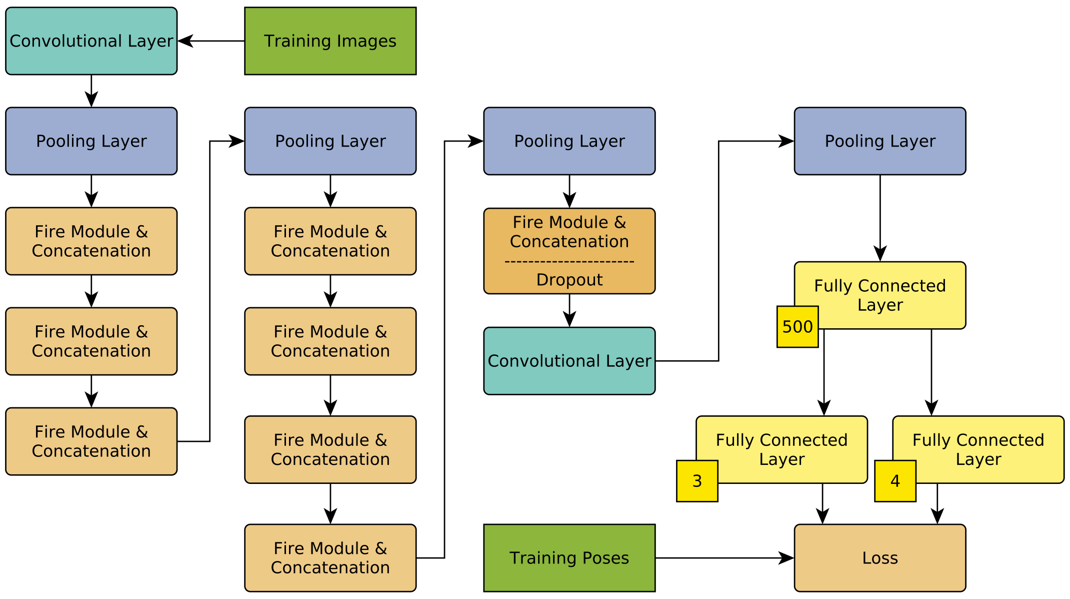

Convolutional Layer
Training Images
Pooling Layer
Fire Module & Concatenation
Fire Module & Concatenation
Fire Module & Concatenation
Pooling Layer
Fire Module & Concatenation
Fire Module & Concatenation
Fire Module & Concatenation
Fire Module & Concatenation
Pooling Layer
Fire Module & Concatenation
Dropout
Convolutional Layer
Training Poses
Pooling Layer
Fully Connected Layer
500
Fully Connected Layer
3
Fully Connected Layer
4
Loss

Figure 5.2: Architecture of SqueezePoseNet. A fully connected layer with 500 neurons is added after the last pooling layer. For pose regression, two fully connected layers, one for the position component with three neurons and one for the orientation component with four neurons is added.

The sum of the position error and the weighted orientation error build the computed loss. $\hat{\mathbf{x}}_i$ and $\mathbf{x}_i$ are ground truth and estimated position in a metric space. $\hat{\mathbf{q}}_i$ and $\mathbf{q}_i$ are ground truth and estimated orientation in quaternions. Since position and orientation are not in the same unit space a weighting is realized by the hyperparameter β to keep the errors terms in the same range. This prevents the CNN to minimize only one of the two error terms with priority. Typically, β is set between 250 and 2000 for outdoor datasets. For the experiments in Section 8.1, β is set to 500.

5.4 Training

CNNs usually need a huge amount of training data to work well, which is often not available. Transfer learning and DA are valuable processes to overcome this issue. While localization supported by DA is addressed in Chapters 6 and 7, the training is supported merely by transfer learning in this chapter. Generally, considering transfer-learning, a network is initially pre-trained with a huge amount of data, which is often publicly available, to determine weights for the network's layers. These weights are used as initial values for later training.

In this case, the original SqueezeNet layers, that are not considered in the modification process, are initialized with weights derived from training on the *ImageNet* dataset. It is shown that training on the *Places* dataset [Zho+16; Zho+14] leads to further improvement in terms of accuracy [KGC15], as it is a more suitable dataset for localization. Whereas the *ImageNet* dataset consists of images showing animals, vehicles and other objects, the *Places* dataset is mainly showing architectural classes as buildings and generally man-made structures. The training is therefore subsequently carried out on the *Places* dataset.

Prior to starting training on the actual *Atrium* dataset (Section 4.1) for evaluation, the CNNs are trained on a localization benchmark dataset, the *Shop Façade* dataset (Section 4.3). Only the weights of the modified layers are updated within this training process, since the preceding layers are already initialized with weights obtained from the pre-training. The network is then fine-tuned on the *Atrium* dataset consisting of 864 images. The inputs for the CNNs are 757 training images and 95 evaluation images with their corresponding camera poses. The training errors on SqueezePoseNet are 4.45 m for position and 14.35° for orientation.

For the evaluation of SqueezePoseNet concerning accuracy, the deeper VGG16-Net, which should tend to be more accurate is considered. The VGG16-Net again is built to solve classification tasks. It is modified in a similar manner as SqueezeNet is modified as described above to estimate poses. The network's layers are initialized with weights obtained by training on the *Places* dataset. The adaptation is trained on the *Shop Façade* dataset to obtain initial weights for the added layers. Therefore, the modified network is trained on the *Atrium* dataset. The training errors on the modified VGG16-Net result in 2.35 m and 9.09° for position respectively orientation. All experiments on the trained networks for camera re-localization are carried out in Section 8.

The workflow can be summarized as follows: (i) Selection of a suitable CNN for a classification tasks. (ii) Initialization of the CNN with pre-trained weights. (iii) Transfer learning on a dataset more suitable for localization, here: *Places* dataset. (iv) Modification of the CNN to solve camera re-localization. (v) Training of the network on the actual training data for evaluation, whereas the trainable layers are restricted to the last fully connected layers. (vi) Evaluation of the CNN on the evaluation dataset.

The training procedure is carried out on a 64 GB RAM computer equipped with an Intel® Core^{TM} $i7 - 3820$ 3.6 GHz processor and an 8 GB GeForce GTX 980 graphics card.

6 Localization with Data Augmentation by Image Rendering

Visual localization is progressively solved by data-driven deep learning methods or hybrid approaches, which combine hand-crafted image analysis and data-driven learning. The performance of data-driven methods highly depends on the underlying training datasets and their characteristics. The characteristics of datasets, which CNNs are trained on, are jointly responsible for the resulting behavior of a trained network. An exemplary behavior of such is proven to be changed from CNNs focusing on texture of objects to focus on the shape of objects instead [Gei+18]. Such behaviors of CNNs are driven and can be actively driven by the user from the choice of the characteristics of the training data and their distribution. Therefore, it is of interest how DA can improve such data-driven learning for network-based pose regression. DA is widespread in the computer vision community to boost performance of data-driven methods. In this chapter, image rendering for DA to boost the performance on camera re-localization tasks is presented. 3D models are generated from training datasets and utilized to create synthetic images from novel views to augment an existing training dataset. Camera re-localization pipelines are trained on these augmented datasets and evaluated in various experiments. The 3D models are thus not utilized directly during runtime, giving the advantage to not provide them online which could be cumbersome for applications running on mobile devices with low storage capacity. In this chapter, improvements of data-driven end-to-end learning for camera re-localization supported by DA with image rendering are investigated.

Parts of this chapter have been published in the following conference paper respectively journal article:

M. S. Müller and B. Jutzi. UAS Navigation with SqueezePoseNet - Accuracy Boosting for Pose Regression by Data Augmentation. *Drones*, Vol. 2, 1, pp. 7–27. 2018.

M. S. Müller, A. Metzger and B. Jutzi. CNN-Based Initial Localization Improved by Data Augmentation. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences – TC I Midterm Symposium "Innovative Sensing – From Sensors to Methods and Applications"*, Volume IV-2/W3, pp. 117–124. 2018.

This chapter is organized as follows. The following section introduces to image rendering and DA. After reviewing the related work in Section 6.2, visual localization by CNNs and feature matching is outlined in Section 6.3. Experiments for quality assessment are carried out in Section 6.4. All experiments on camera re-localization are depicted in Chapter 8.

6.1 Introduction

In this chapter, the problem of camera re-localization by image pose estimation in six DoF is addressed whereas the focus is set on DA by image rendering to improve such localization. DA is especially introduced to investigate the benefit of rendered images in the training process. Therefore a 3D model of the considered scene is utilized to render images from arbitrary viewpoints. The topics of localization and generally CNNs are addressed previously in the Chapters 2.3 and 3.2, whereas DA and 3D models are addressed in the following paragraphs of this section. For quality assessment (Section 6.4), feature matching and image retrieval are carried out and briefly introduced in the following.

Data Augmenation could potentially improve camera re-localization processes. Improving the performance of data-driven methods like CNNs for general computer vision tasks by augmenting the underlying training datasets is widely known and well established. Such DA includes the modification of existing training data as well as the simulation of purely new data to expand training datasets. Common methods include shifting, scaling, rotating, flipping, cropping, compressing or blurring the training images to generate new images that extend the training database with new characteristics. In this thesis purely new images are rendered in the target scene to augment a training dataset. Therewith, training is carried out on a set of training images enriched with synthetic images. CNNs and other data-driven methods benefit from a high distribution of training data. The more varying the representations in the training samples, the more robust and accurate can a model be optimized. Therefore, DA is introduced to overcome recent drawbacks on camera re-localization.

3D Models are utilized for the demands on DA in this thesis. The amount of 3D models or 3D city models increased in the last years and covers large parts of the globe nowadays. Simultaneously, such models became more realistic concerning geometry and texture, and are updated more frequently. Taking this into account, navigation methods for vehicles and pedestrians could utilize this knowledge of the scene for localization and subsequent for navigational purposes or alike. It is underscored, that a 3D model is not utilized directly for the demands of localization in this thesis. That would presume to provide the 3D model at runtime, which conflicts with real-time processing on mobile platforms. Fast computations are mandatory for navigation frameworks on small UAVs or MAVs due to limited computational processing power. Rather, a 3D model is utilized to render additional images for offline training. During application time, merely forward passes are carried out, which can be computed fast. Therefore, a navigation application could run on a small on-board device, supporting autonomous navigation for mobile platforms. The reconstruction of the 3D scene is realized by only accessing the already existing training data. Therefore, no additional data is necessary for the approach on DA by image rendering.

Feature Matching is a fundamental task in image analysis and widely used for different kinds of applications. Local features are extracted and characterized by their descriptors at distinct locations of an image. These descriptors can be compared and matched according to their similarity over multiple images. Therewith, a relation between multiple images can be determined. A lot of computer vision tasks utilize feature matching, e.g. classification, segmentation, detection, image retrieval, 3D reconstruction, tracking methods or image alignment.

Image Retrieval is the task of finding the most similar image in a set of images given a query image. One of such image retrieval methods is CBIR, where colors, shapes or textures of an image are analyzed by computer vision tasks to find similarities between two or more images. In this chapter feature extraction followed by histogram intersection is carried out to find the most similar training image(s) given a query image. In this case, poses are provided for each training image. Thereby, a pose for an evaluation image can be determined by simply assigning the pose of its nearest neighbour or more complex variants like the weighted pose of multiple nearest neighbours. Hence, image retrieval is not merely used to find similar images, but to compute distinct image poses. As depicted above, training datasets are extended by rendered and translated images to increase the quantity and distribution of provided images and poses. An increased number of poses and a denser distribution of such, potentially increase the localization accuracy of a query image by image retrieval. Experiments on image retrieval with DA are carried out in Section 7.4.2.

6.2 Related Work

The focus on related work is set on methods of DA and how 3D models are utilized to support such augmentation. Since feature matching and image retrieval are utilized for quality assessment in Section 6.4, related work on this subjects is also covered in the following. Related work on localization and CNNs is tackled previously in Chapter 5.

Data Augmentation is a well established technique in computer vision [Gha+16; LBC17]. It is shown to boost performance in fields of classification [Tu05; Kar+14; Ng+15], segmentation [Goy+17], object recognition [MS15], object detection [Pen+15], hand gesture estimation [Mol+15] or human pose estimation [RS16]. DA further supports data-driven methods like CNNs to handle invariance which helps to generalize and further boost accuracy [PVZ15; CGK15]. Recently, general DA using GANs showed promising results [SN18]. Furthermore, augmenting training data by generating synthetic images is a valuable process of DA. Synthetic images of text in clutter were generated to train a Fully-Convolutional Regression Network [GVZ16]. Data augmentation regarding camera re-localization is carried out using a CNN to estimate depth values for RGB images followed by synthetic viewpoint generation to enhance the training data [NB17]. This data augmentation in 3D space leads to additional pose coverage and furthermore improves localization accuracy.

3D Models have a high potential to serve for DA. Synthetic images rendered from 3D models have long been used in computer vision to generate extensive training data [SGS10; MSN05]. Rendering images from 3D objects is also practiced to expand training data and improve performance of CNNs [Su+15; Gup+15]. 3D models support the learning process for deep object detectors [Pen+15] or serve for DA for segmentation [Goy+17]. Furthermore such models are utilized to augment datasets for dense 3D object reconstructions [Yan+18] or human 3D pose estimation [RS16]. It is also shown that CNNs trained on synthetic images generalize well to captured images [RS16]. In addition hand gesture estimation is also supported by DA with 3D models [Mol+15; LA17]. The 3D model of a target scene can be utilized to render images for the demands of training localization networks. Reconstructing 3D models is of high interest in researches communities like photogrammetry, computer vision or geo-information sciences [SJ06; PY08; Iva14].

3D models or images with known six DoF poses are the basis to train CNNs for camera re-localization. Such camera re-localization by pose estimation with CNNs is limited by the coverage and possible lack of training data regarding the target scene. It is shown, that pose regression in areas with less training data scores worse compared to pose regression in areas with a dense distribution of training samples (Chapter 5). Utilizing 3D models to overcome lack of training data by DA has high potential. Such models are often available for city scale scenes. In addition, they are simple to reconstruct with automatic and open source SfM and MVS pipelines. Research continuously focuses on the reconstruction of such models and its automation [SJM13; SJ06; Pol+00]. Moreover, recently various benchmark datasets for visual localization [KGC15] and with varying conditions are published [Sat+18a].

Feature Matching is a core task in computer vision and is applied to numerous applications. Preceding to feature matching is the detection and description of such features. There are several established algorithms in this context like SIFT, SURF, BRISK or ORB. Computer vision tasks like image classification [BZM06], object detection [LZ13], tracking [ZYS09], 3D reconstruction [SF16], SLAM [MMT15] or visual localization [SLK16] can be tackled by the support of these algorithms. It is shown that rendering images in point clouds created by laser scans and images improved feature matching and visual localization [Sib+13].. However, in contrast to their work, all 3D models generated or utilized within this thesis are reconstructed automatically only utilizing image data and are therefore less detailed. Techniques considering laser scans are not applicable in this case. Aerial images are matched to terrestrial images by the support of rendered images of a 3D model [Sha+14]. The quality of the rendered images is in this case compensated by rendering from wide distances.

Image Retrieval is an efficient solution to find similar images from databases and became popular with the emergence of large-scale image collections. CBIR considers colors, shapes or textures to associate a query image to its most similar image(s) in an image database. There are several approaches present for image retrieval. Solutions utilize grey values [SM97], Eigenfeatures [SW96], Vector of Locally Aggregated Descriptors [Jég+10] - a compact descriptor to make image retrieval more efficient concerning runtime and storage [AZ13] - or CNNs [Sha+15; BL15]. Image retrieval is improved for situations where the scene appearance changes due to variable illuminations over time by generating virtual views from Google street-view panoramas [Tor+15]. In contrast to this thesis, only individual depth maps are used and no global 3D model.

6.3 Methodology

This section covers the methodology on the utilized camera re-localization pipelines as well as the image rendering for DA on the *Puzzle* and *Atrium* datasets. DA by image rendering and the training processes are described in Section 6.3.1. As part of quality assessment for rendered images, feature matching and image retrieval are applied on the *Atrium* dataset and described in Section 6.4.

For investigating DA, three different CNNs and two datasets are utilized. The investigation leads with an initial experiment on the *Puzzle* dataset, a small sized dataset that serves for feasibility of the approach of DA by image rendering. PoseNet and the modified VGG16-Net introduced in Chapter 5 serve for the subsequent experiments. The complex outdoor dataset of the atrium serves for further experiments. SqueezePoseNet and the modified VGG16-Net serve for the experiments on this dataset. All experiments on camera re-localization are carried out in Chapter 8, whereas experiments on DA by image rendering are tackled in Section 8.2.

6.3.1 Image rendering

CNNs demand training on a huge amount of training data to assure robust and accurate performance. The lack of such training data is a major problem in many fields of data-driven learning. These circumstances also apply for pose regression with CNNs. Therefore, transfer learning is priorly applied as a valuable intermediate step to overcome issues of sparse training data (Section 5). Therewith, the localization accuracy and the training duration could be improved. However, even though transfer learning is a valuable process to help convergence and to speed up the training process, drawbacks caused by sparse training data, unfavorable distributed training data or simple lack of training data are not covered by transfer learning. DA by image rendering is introduced to overcome these drawbacks and to improve the accuracy of localization. In this chapter, DA is realized by image rendering. The general idea is to generate additional images to augment the training dataset. Such image rendering requires a 3D model of the considered scene. An arbitrary amount of images can be rendered within such a 3D model. Thus images with purely new poses are generated, which advance not only the distribution of images, that are fed to the network during training but also increase the space of training poses. A wider distribution of training images and a more diverse space of training poses should contain valuable information for data-driven learning. Therewith, CNNs should be able to learn the appearance of images and their poses in a more extended space compared to training on merely the original training data.

The actual rendering process for the *Puzzle* dataset and the *Atrium* dataset is covered in the following. Datasets for localization usually persist of RGB images and their corresponding poses. To generate such data, 3D models, that serve for rendering are reconstructed. For this purpose, PhotoModeler [Pho] – a software to reconstruct 3D models from images – is utilized to generate the geometry and the texture from the *Puzzle* dataset. Due to the geometric simplicity of the scene, which is a single plane, the model is reconstructed with photo-texture with little effort. Since, the textures of the model are created directly from the input images, a photo-realistic model is reconstructed by mosaicking. To render images from arbitrary views in such a 3D model, the camera intrinsics and the poses of these views have to be set. The camera intrinsics are obtained within the 3D model reconstruction and are adopted for the purpose of image rendering. The poses of the new views are generated automatically within the space of the ground truth evaluation poses from the original dataset. The actual rendering is carried out by simulating a virtual camera with the obtained intrinsic parameters in the virtual scene, consisting of the reconstructed 3D model. The simulation and rendering is carried out with *gazebo* [KH04; Fou], a software for simulating sensors and robots.

The rendered images appear accordingly realistic compared to the captured images, as can be seen in Figure 6.1. The Figure shows exemplary two captured images in the top row and their corresponding rendered images in the bottom row. The pose of Figure 6.1 (a) is determined incorrectly within the reconstruction process, leading to an incorrect rendered image shown in Figure 6.1 (c). The pose of Figure 6.1 (b) is determined correctly, whereby the corresponding image in Figure 6.1 (d) is rendered accordingly correct. Images with wrong determined poses are not added to the dataset nor utilized for further processing. Overall, 479 rendered images are generated to augment the original dataset of 19 captured images. It is highlighted, that no additional data beside the original training data nor any assumptions are necessary to realize this DA approach.

Concerning the *Atrium* dataset, image rendering is carried out slightly different compared to the image rendering for the *Puzzle* dataset. The atrium is a scene with more complex geometry, rather than a single plane. Therefore, the 3D model is reconstructed with Agisoft [Agi17] – a SfM software. The triangulated mesh is colored by the RGB information from the dataset images. The camera intrinsics are obtained within the 3D model reconstruction and are adopted for subsequent rendering. Multiple augmented sub-datasets are created for the experiments on the *Atrium* dataset. Therefore, poses for the new views are generated to serve for rendering additional images. In total, four sub-datasets with the following properties are generated. One dataset is augmented with rendered images from views with the exact poses as the ground truth evaluation poses. A second dataset is augmented with rendered views from poses generated near the ground truth evaluation poses. Therefore, a translational and rotational offset is added to each of the original evaluation poses to generate the new poses. Such offset is added to each pose individually, whereas the offset corresponds to a Gaussian noise, generated with a mean of 0 and a σ of 1 m for the translational component and a mean of 0 and a σ of 0.1 for the rotational component. The dataset that consists of rendered images with the same poses as the evaluation images is named *Coincide*. The dataset with the noise offset on each pose is named *Diverge*. Two

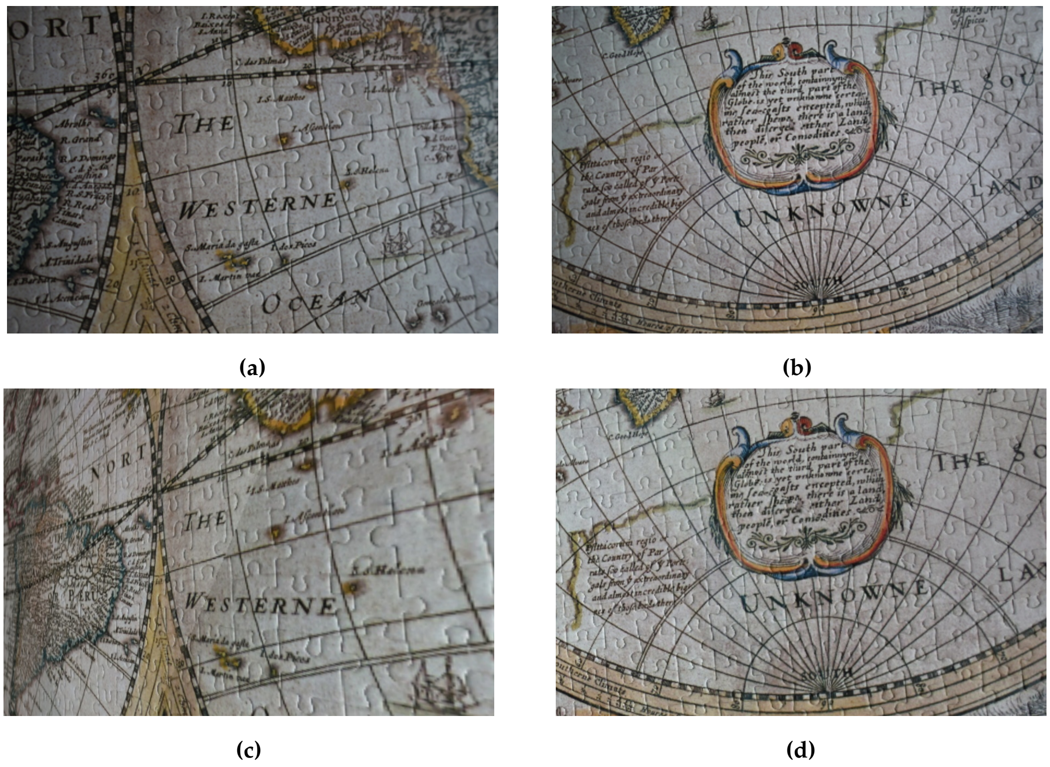


Figure 6.1: Captured images of the *Puzzle* dataset (top row) and their corresponding rendered images (bottom row). The pose of the captured image (a) is determined incorrectly within the reconstruction process leading to an erroneous rendered image (c). Images with incorrect poses were disregarded in all subsequent processes. The pose of image (b) is determined correctly, whereby the corresponding rendered image (d) is accordingly correct.

further datasets are created by a combination of the original dataset consisting of captured images and the datasets with rendered images. The captured images combined with the *Coincide* dataset form the *Captured+Coincide* dataset. Likewise, the *Diverge* dataset, combined with the captured images from the original dataset form the *Captured+Diverge* dataset. The actual rendering of the images is carried out by simulating a virtual camera with the intrinsics obtained within the SfM process. The rendered images of the *Atrium* dataset appear not as realistic as the ones from the *Puzzle* dataset. This is a result of the more complex scene and the 3D model consisting of a high number of surfaces affecting geometry and texture similarly. Figure 6.2 shows exemplary two captured images in the top row and their corresponding rendered images in the bottom row. Since the rendered images are generated with the camera model of the captured images, the corresponding images have the same scene view. Diverse camera models could technically be chosen to enrich a training dataset with even more various images if desired.

To show the benefit of the utilization of rendered images, training is carried out with the original *Puzzle* training dataset and the augmented *Puzzle* dataset as well as with the original *Atrium* dataset and the augmented *Atrium* dataset. Regarding the *Puzzle* dataset, the evaluation is realized on two CNNs, PoseNet and the modified VGG16-Net. The focus is set on relative performance improvements of the network's accuracy. The training process with the original *Puzzle* dataset – without any rendered images – by PoseNet

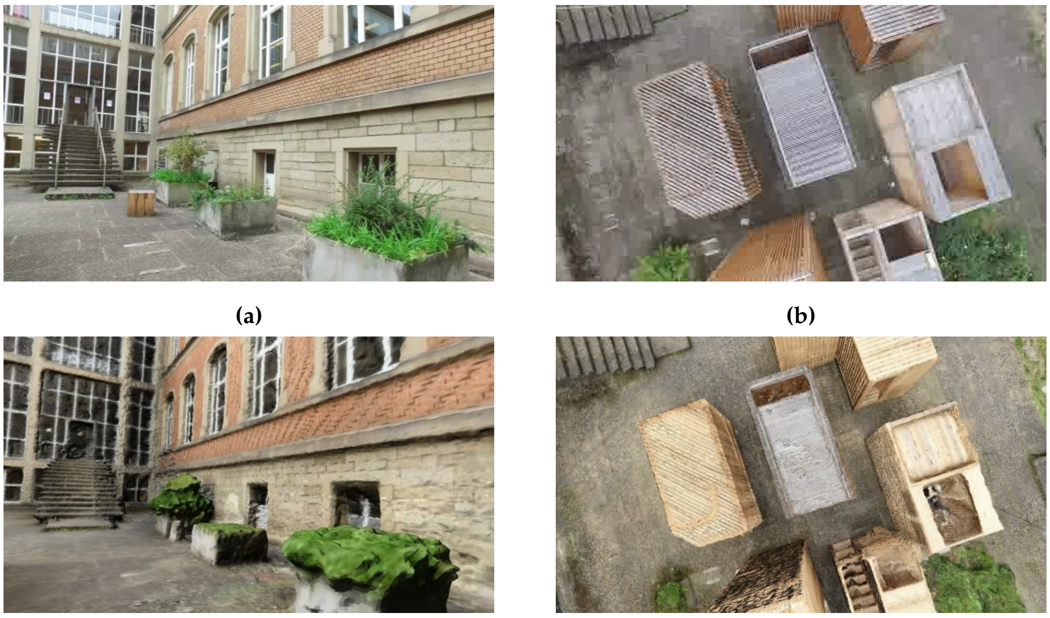


Figure 6.2: Captured images of the *Atrium* dataset (top row) and corresponding rendered images (bottom row). The rendered images are generated utilizing the same camera model as the captured images. Diverse camera models could technically be chosen to enrich a training dataset with even more various images if desired.

ended in training errors of 0.13 *m*, 6.00° The training of the modified VGG16-Net ended with training errors of 0.20 *m*, 5.00°. The training with the added 479 rendered images ended for PoseNet with a training error of 0.18 *m*, 6.44° and for the modified VGG16-Net with 0.04 *m*, 1.98° respectively.

Regarding the *Atrium* dataset, training is carried out on the *Coincide* and *Diverge* datasets with initial weights obtained by transfer learning on the *Shop Façade* data. The model weights of the *Captured* dataset are furthermore used to initialize the networks for training on the *Captured+Coincide* and *Captured+Diverge* datasets for faster convergence. Concluded, the datasets of *Places* and *Shop Façade* are utilized for transfer learning to preserve initial network weights. For the *medium* and *low coverage* set training is carried out on the following training dataset types. *Captured*, *Coincide*, *Captured+Coincide*, *Diverge* and *Captured+Diverge*, whereas a single dataset of the dataset type *Captured* is utilized for training on the *medium* and *low coverage* set. For each of the other dataset types, two distinct datasets exist corresponding to each of the two evaluation sets. Therefore, nine different training datasets are utilized. A more detailed description of the datasets is denoted in Section 4.1. The CNNs are evaluated on the *medium* and *low coverage* set in Chapter 8.

6.3.2 Tasks for Quality Assessment

The challenging conditions of the *Atrium* dataset are highlighted by feature matching and image retrieval in Section 6.4. The methodology to these tasks is briefly introduced in the following.

Feature Matching

A prerequisite for many computer vision tasks is feature matching. Considering DA, such feature matching is investigated for comparison to the introduced data-driven end-to-end approaches. Implicitly, feature matching is investigated based on the number of inlier matches between images from training datasets to images from an evaluation dataset. To save computational effort and avoid matching every evaluation image to each single training image of the training dataset, Bag of Visual Words (BoVW) is applied. The goal is to compare a single evaluation image to a data base of training images. A visual vocabulary is created by utilizing SURF to extract features and descriptors from every training image. Based on the visual words, a histogram for every training image is derived. Subsequently the features and descriptors of the evaluation images are derived. Adjacent, a histogram of visual words of the evaluation image is derived and compared to the BoVW by using histogram intersection. Therewith, the best matching images are obtained and classical feature matching between every evaluation image and its top three closest training images is performed. As a measure of quality the number of inlier matches between evaluation images and training images is taken into account.

Image Retrieval

A retrieval pipeline based on 3D color histograms to describe each image is used for quality assessment [Ros]. These histograms are extracted from each image in the training datasets and each evaluation image. Subsequently, histograms from the evaluation images are compared by a similarity measure based on chi-squared distances to all images in the training datasets. The chi-squared measure is used to compare discrete probability functions, which is very suitable for histograms as they depict such probability distributions. As labeled poses exist for each training image and each evaluation image, pose differences can be computed between corresponding images.

$$d = \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2 \ , \tag{6.1}$$

where $\mathbf{x}_i$ is the position of an evaluation image and $\hat{\mathbf{x}}_i$ is the position of a training image. $\|\cdot\|_2$ depicts the Euclidean Norm.

The difference of rotation between an evaluation image and a training image θ is computed by

$$\theta = 2 * arccos\left(\mathbf{q}_i \cdot \frac{\hat{\mathbf{q}}_i}{\|\hat{\mathbf{q}}_i\|_2}\right) \ , \tag{6.2}$$

where $\mathbf{q}_i$ is the normalized quaternion of an evaluation image and $\hat{\mathbf{q}}_i$ is the quaternion of a training image. θ therefore depicts the angle between the orientation of a training and an evaluation image.

6.4 Quality Assessment

The challenging conditions of the *Atrium* dataset are highlighted by carrying out feature matching between the training dataset and the evaluation datasets. The potential benefit of the rendered images within the augmented sub-datasets is analyzed regarding their suitability for image analysis by image retrieval subsequently.

6.4.1 Feature Matching

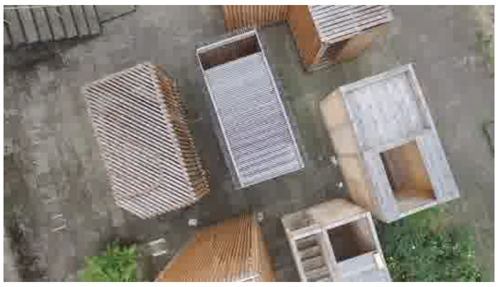
Preceding to the experiments on camera re-localization in Chapter 8, feature matching is applied on the original dataset of the atrium scene to show the difficulties concerning image analysis methods on that particular dataset.

The experiments shall expose the challenges of the evaluation datasets due to low coverage. Satisfying image matching between the training images of the *Captured* dataset and the evaluation data could not successfully be determined due to insufficient number of inlier matches for homography estimation. Explicitly the evaluation images of the *medium coverage* set have on average 152.2 matches between an evaluation image and its assigned nearest training images according to BoVW. After inlier test by RANSAC the confidential matches drop to 6.2 on average. The analogous test on the *low coverage* set shows 120.3 matches per image and 3.4 inlier on average. An overview including additionally the maximum number of matches and inlier is given in Table 6.1.

Feature matching	*medium coverage set*	*low coverage set*
# matches (max.)	254	208
# matches (avg.)	152.2	120.3
# inlier matches (max.)	38	13
# inlier matches (avg.)	6.2	3.4

Table 6.1: Evaluation of feature matching on the *medium* and *low coverage* set. The number of matches and inlier matches between evaluation images and nearest training images according to BoVW from the *Captured* dataset are depicted. The evaluation images are assigned to their nearest training images by BoVW. On average 152.2 respectively 120.3 matches between an evaluation image and its nearest training image could be found. However, the number of average inlier with 6.2 respectively 3.4 is unsatisfying for subsequently image matching, which shows the difficulty of this dataset.

A visualization of an evaluation image of the *low coverage* set and its nearest neighbor from the training dataset determined by BoVW is visualized in Figure 6.3. Due to wide baselines, perspective changes and low coverage sufficient image matching could not be determined. This circumstances appear all over the dataset. The quality assessment by feature matching depicts the difficulty of this dataset and its challenges due to wide baselines between training and evaluation data. Camera re-localization suffers from the same deficiencies in such datasets, wherefore DA is considered to overcome the drawbacks


(a)

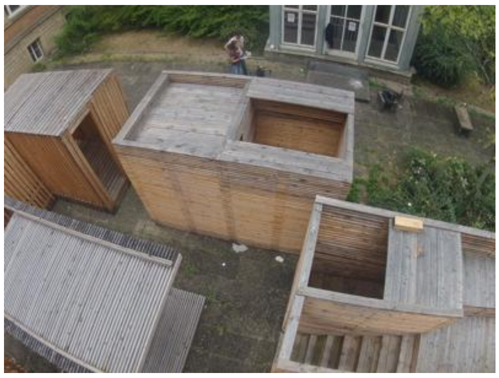

(b)

Figure 6.3: Evaluation image (a) of the *low coverage* set and its nearest neighbour from the training dataset (b) according to BoVW. A feature matching between these two images could not be determined sufficient due to low number of corresponding image features.

of wide baselines by generating additional data for a more dense distribution of training images and poses.

6.4.2 Image Retrieval

Image retrieval is carried out on the original dataset of captured images and all datasets augmented with rendered images. The mean pose differences are computed between each evaluation image and its top 1, top 4 and top 10 nearest neighbours for each training dataset. The results are depicted in Tables 6.2 and 6.3.

	Mean Pose Difference		
Dataset type	Top 1	Top 4	Top 10
Captured	35.3 m, 126.1°	35.0 m, 114.1°	33.5 m, 107.5°
Coincide	**22.4 m, 90.1°**	**22.8 m, 86.6°**	**22.3 m, 84.1°**
Captured+Coincide	24.7 m, 100.6°	24.9 m, 96.0°	24.4 m, 93.7°
Diverge	22.6 m, 97.0°	22.9 m, 94.7°	22.6 m, 93.2°
Captured+Diverge	26.2 m, 107.2°	26.2 m, 101.1°	25.7 m, 97.7°
Improvement	35.5%, 28.5%	34.9%, 24.1%	33.4%, 21.8%

Table 6.2: Mean pose differences of evaluation images to their nearest neighbours from the training datasets for the *medium coverage* set. The mean pose differences are computed between each evaluation image and its top 1, top 4 and top 10 nearest neighbours for each training dataset. Best results are highlighted in bold style.

Augmenting the original training dataset with rendered images clearly leads to better results than using merely the captured images. Image retrieval is benefiting from a denser distribution of training images, hence finding images in the database closer to the query image and thus improving the pose estimate. The translational component is improved significantly on both, the *medium* and the *low coverage* evaluation set by about 33% − 55%. The rotational component regarding the *low coverage* set with about 5% − 29% is not

	Mean Pose Difference		
Dataset type	Top 1	Top 4	Top 10
Captured	22.17 m, 80.8°	21.2 m, 81.0°	20.6 m, 88.2°
Coincide	12.6 m, 93.8°	12.1 m, 91.9°	12.0 m, 94.2°
Captured+Coincide	11.2 m, 80.0°	11.6 m, 80.7°	10.5 m, 83.3°
Diverge	**9.9 m**, 79.8°	**9.6 m**, 78.3°	**9.6 m**, **79.2**°
Captured+Diverge	11.8 m, **76.5**°	11.1 m, **77.1**°	10.9 m, 80.5°
Improvement	55.3%, 5.3%	54.7%, 4.8%	53.4%, 10.2%

Table 6.3: Mean pose differences of evaluation images to their nearest neighbours from the training datasets for the *low coverage* set. The mean pose differences are computed between each evaluation image and its top 1, top 4 and top 10 nearest neighbours for each training dataset. Best results are highlighted in bold style.

improved as much. Concluded, all datasets enhanced by DA score better results than the original dataset without augmentation. The experiments on camera re-localization with DA by image rendering are carried out in Section 8.2.

7 Localization with Data Augmentation by Image-to-Image Translation

In this chapter, image-to-image translation is introduced to enhance the quality of images by mapping them between multiple domains. Generally, image-to-image translation is a constrained synthesis process of mapping an input image to an output image. An image is synthesized with respect to a specified constraint such as another image. In this chapter, the process of DA by image-to-image translation to enhance image analysis tasks is investigated. DA is meant to help overcome limitations and boost performance for further image processing. Parts of this chapter have been published in the conference paper

> M. S. Müller, T. Sattler, M. Pollefeys and B. Jutzi. Image-to-Image Translation for Enhanced Feature Matching, Image Retrieval and Visual Localization. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences - Photogrammetric Image Analysis & Munich Remote Sensing Symposium*. Volume IV-2/W7. pp. 111–119. 2019.

This chapter is organized as follows. The following section introduces the idea of DA by image-to-image translation and selected computer vision tasks that are utilized for evaluation purposes subsequently. After reviewing related work on image-to-image translation, camera re-localization as well as related work on selected tasks for quality assessment which are feature matching and image retrieval in Section 7.2, the methodology is depicted in Section 7.3. Experiments for quality assessment are carried out in Section 7.4. The main experiments on camera re-localization are carried out in Chapter 8.

7.1 Introduction

The performance of data-driven learning methods scales with the quantity and quality of the underlying training dataset utilized for the optimization in the training process. Deep learning with CNNs and GANs enables the development of new data-driven approaches. Both are affected by the characteristics of the utilized training data. With growing interest of deep learning, the demand for suffice and versatile training data is increased and training datasets for numerous tasks are recently published. In this chapter, new training images are generated to augment training datasets by image-to-image translation to investigate the performance of common computer vision and photogrammetry tasks.

Again the focus is set on camera re-localization. The general term for generating new training samples to enlarge given datasets is DA. Augmenting training data is a powerful option to overcome challenges in several fields of computer vision, like feature matching, image retrieval and visual localization. Such DA includes the modification of existing training images as well as the generation of new images to expand training datasets. Common methods in image processing among others are to shift, rotate, scale, flip, crop, transform, compress or blur training images to extend a basis dataset and generate a higher versatility of characteristics within the training data. In this chapter, new images are rendered and furthermore translated by a GAN to augment a dataset of images. Whereas the rendering is carried out in a similar way as described in Chapter 6, the main focus is set on the image-to-image translation. CNNs and other data-driven methods benefit from versatile training datasets. The higher the versatility of training samples provided for the optimization, the more robust and accurate networks can be expected to be trained.

Image-to-Image Translation made a huge leap with the progression of deep learning algorithms and a better understanding of such. Image-to-image translation is the process of translating an images from one domain into another. This could be translations between daytime and nighttime, translations between the four seasons spring, summer, autumn and winter or even the translation of artistically styles. The success of data-driven algorithms like deep learning depends much on the provided training dataset, hence suffering from deficient training data. An insufficient variation of training images weakens the estimates in terms of robustness or accuracy. Typical training datasets for camera re-localization consist of images captured in a specific scene, their corresponding poses and optionally the parameters of the camera intrinsics. Augmenting such training datasets to generate a higher quantity and versatility of training samples over the original dataset has the potential to enhance methods that learn from this data. An augmentation could be undertaken by capturing additional images manually, determining their poses and adding them to an existing training set. However, this is cumbersome and not constructive to tackle the problem of insufficient data. In this chapter, existing training data is augmented by synthetic images. This is carried out by rendering new images as in Chapter 6 and furthermore mapping them from their *rendered* domain to a *captured* domain to provide higher similarity to the evaluation data. The rendered images are generated by utilizing only the pre-existing captured data of the original training dataset. There is no necessity for further assumptions or manual capturing of new data. Given a dataset consisting of images and their corresponding poses of a specific scene, a 3D model of the scene is generated by utilizing a SfM pipeline. Images with arbitrary poses are rendered in this model and used to enhance the original training dataset. However, the rendered images differ strongly in appearance from the original captured training images since the 3D model is no photo-realistic representation of the scene. Since the direct utilization of such rendered images may not suffice as training data for further applications, image-to-image translation is applied subsequently. By image-to-image translation, the rendered images are transformed from their *rendered* domain into the *captured* domain, thus enhancing their appearance to be more realistic. Therefore, the translated images have a higher similarity to the originally captured images including the evaluation images. The higher similarity to the original training images increases the feasibility for potentially serving as additional training data. As for image rendering, the image-to-image translation approach requires no need to capture new data or to make any assumptions. The image-to-image translation pipeline is trained only on the original training dataset and the rendered images. In summary, image rendering is combined with image-to-image translation for DA to enhance common computer vision tasks with focus on camera re-localization. The evaluation of the newly generated training data, namely the images translated from the *rendered* domain into the *captured* domain, is carried out by performing common computer vision tasks. In detail, feature matching and image retrieval is performed as a part of quality assessment in this chapter (Section 7.4). Experiments on camera re-localization are carried out in Chapter 8 to investigate the beneficial impact of image-to-image translation.

Camera Re-Localization is the task of determining a camera pose of one or multiple query images in a specific scene. Camera re-localization carried out using CNNs steadily improved in terms of accuracy over the last years. In general, CNNs are trained on training datasets containing images of a scene and their corresponding poses. The neural network is optimized by iteratively updating the model weights with respect to a loss function. For camera re-localization, this loss function is often based on pose differences or reprojection errors. Camera re-localization could benefit by expanding the training datasets with a more versatile distribution of images and poses. Again, these training datasets are extended by utilizing a 3D model to render new images and apply image-to-image translation to transform rendered images into a more realistic *captured* domain. All experiments on camera re-localization are carried out in Chapter 8, whereas DA by image-to-image translation is tackled in Section 8.3 explicitly.

7.2 Related Work

DA is well established in the field of computer vision to augment training datasets. DA boosts performance in classification [Ng+15], segmentation [Goy+17], object recognition [MS15], object detection [Pen+15], hand gesture estimation [Mol+15], camera pose regression [MMJ18] or human pose estimation [RS16]. Data-driven methods like CNNs can be trained to improve handling invariances like translation or rotation which helps for generalization of such networks [PVZ15]. An augmentation process is introduced, where a network generates augmented data during the training process of a target network to reduce that networks loss, showing improvements on several datasets [LBC17]. Furthermore augmenting training data by synthesizing completely new images is known as a valuable process of DA. Synthetic images of text in clutter are generated to train a Fully-Convolutional Regression Network [GVZ16] successfully.

Image-to-Image Translation refers to a synthesis process where an input image is mapped to an output image. Approaches considering paired training data, where images of the source and target domain are provided in corresponding pairs have been addressed for translations between domains like *grayscale* and *color* [ISI16], *day* and *night*, *aerial* and *map* and others [Iso+17]. One of these networks, pix2pix [Iso+17] utilizes conditional GANs, where an additional conditioning input information is provided for the generator and the discriminator during the training process. However, paired training data does often not exist and is cumbersome to generate. To address this issue, an unpaired image-to-image translation framework is proposed, called cycle-consistent GAN or CycleGAN [Zhu+17]. CycleGAN consists of two separate GANs, where one network maps an image from one domain into another and the other network provides the inverse translation. The two GANs are jointly trained adding a cycle-consistency loss that considers the output of both networks simultaneously. Image-to-image translation trained on unpaired data, has been addressed for artistically style transfer [JAF16; GEB15] or other domain translations like *horse* to *zebra*, *summer* to *winter* or vice versa [Zhu+17]. Such image-to-image translation showed beneficial impact for feature matching and image retrieval translating nighttime to daytime images [Ano+19; PMN18]. With recent research, the number of domains is extended to numerous, e.g. 16 translations between artistically styles or four domains for translations between the seasonal domains as *spring*, *summer*, *autumn* and *winter* [Ano+18]. These translations are predominantly carried out utilizing GANs [Goo+14]. Adversarial networks are also used to generate training data by transforming rendered images of eyes to more realistic samples for eye gaze estimation [Shr+17a].

Camera re-localization by pose regression with CNNs is firstly solved with the publication of PoseNet [KGC15]. Further development of loss functions [KC17] or the implication of Long-Short Term Memory [Wal+17] boosted performance of camera re-localization. Other research focuses on transferring pose regression from large to small networks reducing memory requirements [MUJ17] (Chapter 5). DA is tackled by adding rendered images to the training dataset to improve performance of a pose regression pipeline [MJ18] (Chapter 6). Latest developments are combining deep learning and the well-studied PnP problem [Gao+03] to regress six DoF poses from images [BR18]. The pipeline applies DSAC, a framework of a differentiable RANSAC for finding 2D-3D matches, followed by a pose hypothesis estimation. This approach scores similar results as feature-based methods on camera re-localization. The determination of image poses in challenging scenes with changing weather, lighting or seasonable conditions is important for the navigation of self-driving vehicles and the localization for augmented-reality applications. Therefore, datasets covering these characteristics are published recently [Sat+18a]. Another work, focuses on matching paintings and historical photographs to a 3D model for pose estimation, whereas features are learned to match between paintings and rendered images [ARS14].

Considering the general camera re-localization pipeline introduced in Chapter 2, the focus on improving the final pose estimate is tackled by improving the DA module. Therefore, the rendered images are adjusted by image-to-image translation to fit the target domain in this chapter.

7.3 Methodology

The focus is set on image-to-image translation and the workflow to augment a training dataset with translated images in this section. Experiments for quality assessment are carried out in Section 7.4, where DA for feature matching and image retrieval is investigated. All experiments on camera re-localization by data-driven learning are carried out in Chapter 8.

A general workflow of how image-to-image translation is employed to translate images from a training dataset of captured images and embedding them into a localization pipeline is shown in Figure 7.1. A training dataset containing captured images (① in Figure 7.1)

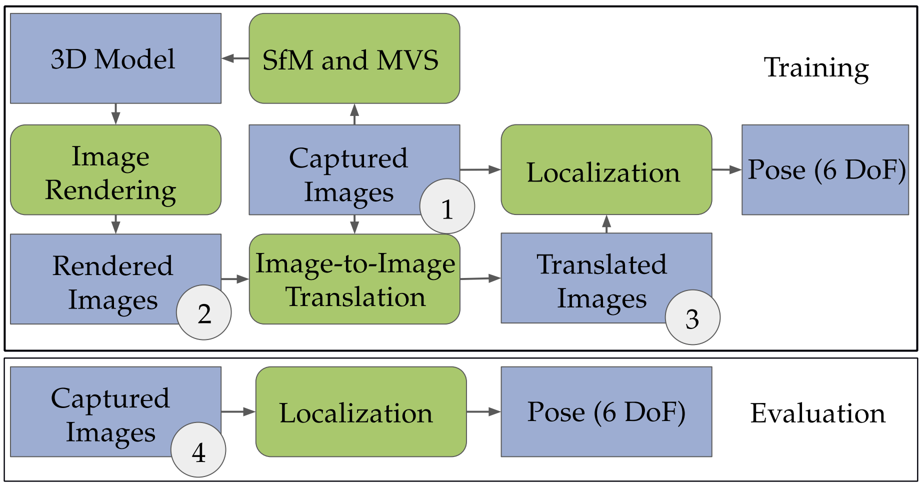


Figure 7.1: A general workflow of how image-to-image translation is employed to translate images from a training dataset of captured images and embedding them into a localization pipeline. All data is depicted in blue boxes whereas methods are depicted in green boxes. Captured training images ① are used to reconstruct a 3D model through SfM and MVS. Rendered images are generated with arbitrary poses and camera intrinsics within this 3D model ②. These rendered images and the original captured images serve as input to train an image-to-image translation network. This network translates images in the *rendered* domain into translated images in the *captured* domain ③. Captured ① and translated images ③ are used to train a localization pipeline. Training is also carried out with merely rendered images ②. All experiments are carried out on the captured images from an evaluation set ④. None of the evaluation images is utilized in the prior training process. For comparison purpose each experiment is carried out on the captured images, the rendered images and the translated images in Section 8.3. It is highlighted, that all images in the *rendered* ② and *captured* ③ domain are generated by utilizing merely the captured images ① from the original dataset.

serves to reconstruct a 3D model through SfM and MVS [SF16; Sch+16]. The model is used to render images with arbitrary poses and camera intrinsics ②. These rendered images and the original captured images serve as input for training an image-to-image translation network. The trained network then translates images from the *rendered* domain to the *captured* domain ③. Captured ① and translated images ③ are used to train a localization pipeline. Training is also carried out with merely rendered images ②. All experiments are carried out on the captured images from an evaluation set ④. None of the evaluation images is utilized in the prior training process. For comparison purpose, each experiment is carried out on the captured images, the rendered images and the translated images in the experiments chapter (Chapter 8). The *Shop Façade* dataset (Section 4.3) from the *Cambridge Landmarks* benchmark [KGC15] serves for these experiments. The scene mainly shows two façades of a shop. The training dataset consists of 231 images and their corresponding poses, whereas the evaluation dataset consist of 103 images and poses. Quality assessment regarding feature matching as well as image retrieval is carried out in Section 7.4.

7.3.1 Image-to-Image Translation

Image-to-image translation is carried out utilizing ToDayGAN [Ano+19], a GAN based on CycleGAN [Zhu+17]. GANs generally consist of two independent neural networks which compete with each other. A so-called generative network generates synthetic images while a discriminative network tries to distinguish the output of the generative network between real and synthetic data. This procedure allows to generate a vast amount of synthetic data while retaining a realistic appearance and thus potentially serves for DA. The fundamentals of GANs are described in-depth in Section 3.3. An image-to-image translation network computes a mapping of images between two domains C and R, corresponding to the *captured* and *rendered* domain. Unpaired samples of both domains c_i and r_j, where $i = 1, ..., N$ and $j = 1, ..., M$ are provided during training. An alignment of training samples is not necessary due to the cycle consistency loss introduced with CycleGAN. The network consists of two generators $G_R : R \rightarrow C$ and $G_C : C \rightarrow R$ to translate images between the domains as well as two discriminators D_C and D_R to distinguish between translated and captured images (Figure 7.2).

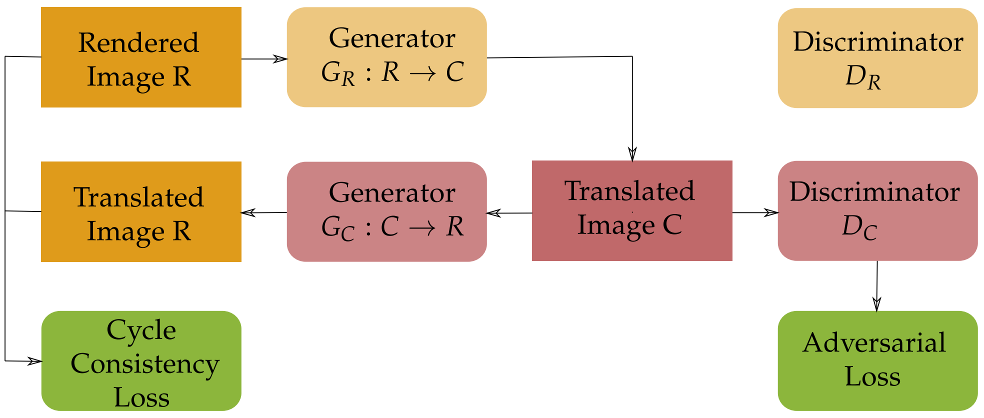


Figure 7.2: Overview of the utilized GAN system. The training pass for the direction from a rendered image to a captured image $R \rightarrow C$ is shown. Besides the adversarial loss, a cycle consistency loss is utilized to encourage inverse mappings such that $G_R(G_C(c)) \approx c$. The training pass for the opposite direction $C \rightarrow R$ is executed likewise. Discriminator D_R is illustrated for completeness.

The cycle consistency loss specifies the constraint in such way that a translation $R \rightarrow C$ followed by $C \rightarrow R$ is hold to lead to the same image as the original input image.

$$G_R(G_C(c)) \approx c \tag{7.1}$$

For the purpose on augmenting the *Shop Façade* dataset, rendered images are translated from the *rendered* domain to the *captured* domain. Therefore, the rendered-to-captured generator $G_R : R \rightarrow C$ is used. The training images from the *Shop Façade* dataset serve as training samples for the *captured* domain. Images rendered from multiple poses in the 3D model of the scene serve as training samples for the *rendered* domain. The 3D model is reconstructed using COLMAP. Poses for rendering additional images are generated in a grid with a spacing of 25 *cm*. Poses are only generated up to three meters away from the nearest original training pose. The orientation of each generated pose is set to the orientation of the nearest training image. Thereby, additional poses have been generated to render images from new positions and with different points of view. In total 2652 images are rendered, which serve together with the 231 captured images for training the image-to-image translation network. Figure 7.3 shows synthetic generated poses of the rendered images (dark red), training poses of captured images (red) and evaluation poses of the captured images (green). Only every tenth of the synthetic generated poses is depicted for visualization purpose in this figure. An exemplary rendered image from the 3D model is illustrated in Figure 7.4 (a). Figure 7.4 (b) shows the image after translation into the *captured* domain by the image-to-image translation network.

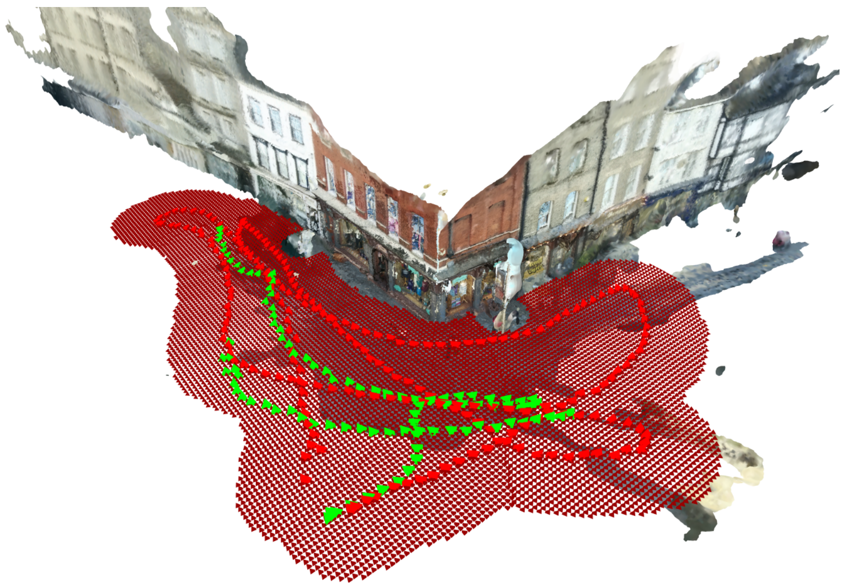
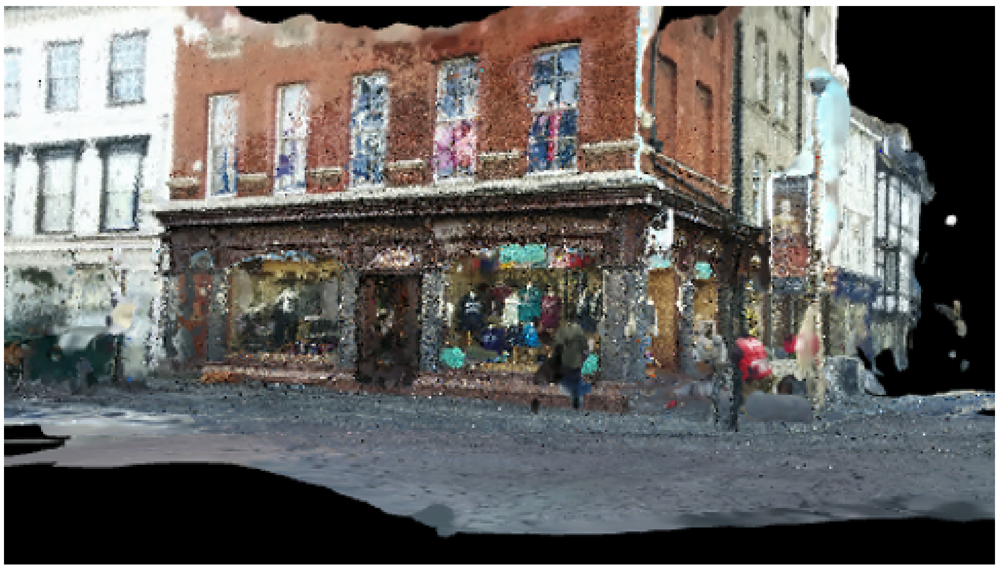


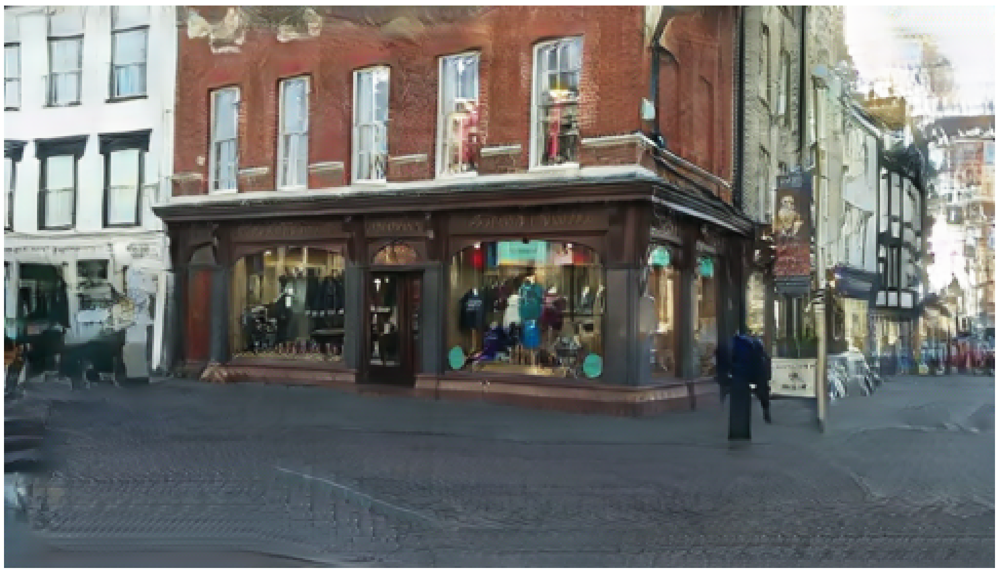
Figure 7.3: Visualization of synthetic generated poses for the rendered respectively translated images (dark red), poses of of the captured training images (red) and poses of the captured evaluation images (green).

7.3.2 Tasks for Quality Assessment

The impact of image-to-image translation on different computer vision tasks – namely feature matching, image retrieval and camera re-localization – is evaluated. Quality assessment by feature matching and image retrieval is carried out in Section 7.4. All experiments on camera re-localization are carried out in Chapter 8.



(a)



(b)

Figure 7.4: Exemplary rendered image (a) and corresponding translated image (b). The translated image is generated by image-to-image translation whereas the rendered image serves as input. The image in figure (b) is rendered from a pose, which does not exist in the original dataset. Hence, additional value is generated with the rendering (of new views) followed by image-to-image translation.

Feature Matching

Feature matching is performed to evaluate the quality of the images generated by rendering and image-to-image translation. The performance of feature matching is measured based on the number of inliers between images from a training dataset and images from the evaluation dataset. Inliers are depicted within a geometric similarity transformation [HZ03] while a variant of MLESAC [TZ00] for model fitting is applied. Feature detection and description is carried out using SURF. The actual feature matching is performed by an approximate nearest neighbour search [ML09]. To ensure matching by an overlapping field of view between evaluation images and training images, a BoVW [Csu+04] approach is deployed. Concerning feature matching, according to a query image the ten nearest neighbours in the training set are considered. As a measure of quality, the number of inliers between evaluation images and their nearest training images are taken into account, whereas a high number of inliers corresponds to a high quality. Feature matching is performed on the three training datasets of the *captured*, *rendered* and *translated* domain in Section 7.4.1.

Image Retrieval

For further evaluation of the feasibility of image-to-image translation, image retrieval is applied using a BoVW approach. The goal is to compare single evaluation images as query images to a set of training images and to find the images with the highest similarity. Subsequently, a pose difference is computed by taking the pose of the evaluation image and the poses of the most similar training images into account. A localization of evaluation images is realized that way. Therefore, a visual vocabulary with 250 visual words is created by utilizing SURF to extract features and their descriptors from all training images. All features are clustered using k-means [Llo82] with 250 clusters, whereby every cluster represents a visual word. Based on these visual words a histogram for every training image is derived. Subsequently the features and descriptors of the evaluation images are derived with the same strategy and added to one of the 250 clusters by using a nearest neighbour approach. Adjacent, a histogram of visual words of the evaluation image is derived and compared to the BoVW by using histogram intersection. Therewith, the best matching histograms of the images from the training set are identified and assigned to an evaluation image. The images corresponding to these histograms are considered as the nearest neighbours for the evaluation image. To evaluate image-to-image translation in Section 7.4.2 the performance of image retrieval on the three datasets, which are captured images, rendered images and translated images, is investigated. Besides a visual comparison, a numerical evaluation is carried out as mentioned by computing pose differences between ground truth and estimated poses. The ground truth poses for training and evaluation datasets are given from the *Shop Façade* benchmark dataset. The estimated poses again are derived by determining the mean poses of the nearest training images as introduced in Chapter 6.

Camera Re-Localization

For further investigation on the feasibility of translated images to enhance image analysis, camera re-localization is performed utilizing DSAC++ [BR18]. This approach consists of a neural network and a pose estimation pipeline based on 2D-3D correspondences. The network takes RGB images, their corresponding poses and intrinsic camera calibrations as input for the training procedure and regresses a six DoF pose for single evaluation images. Initially a CNN predicts scene coordinates for image patches from a given input image. This leads to 2D-3D correspondences between pixels and their corresponding point in the 3D scene from predicted scene coordinates. By solving the PnP problem, a camera pose is estimated from such 2D-3D correspondences. Multiple camera pose hypotheses are computed – each from four of such 2D-3D correspondences. This is followed by a pose hypothesis selection and a pose hypothesis refinement leading to a final pose estimate. The network is optimized by minimizing a pose loss in an end-to-end training using backpropagation. The training sets consist of the captured images from the *Shop Façade* dataset, the rendered images generated from a 3D model and the translated images generated by image-to-image translation.

The experiments on DA by image-to-image translation for camera re-localization by data-driven methods are carried out in Section 8.3 in the experiments chapter.

7.4 Quality Assessment

For quality assessment of the images generated by rendering and image-to-image translation feature matching and image retrieval are carried out in this section preceding to camera re-localization in Chapter 8. All of these experiments are carried out on the *Shop Façade* dataset from the *Cambridge Landmarks* visual localization benchmark.

7.4.1 Feature Matching

The improvement of feature matching on translated images is investigated in contrast to feature matching on captured and rendered images. Therefore, SURF features are extracted from all training images and evaluation images. The training images include the original captured training images, the rendered images and the translated images. Since matching every evaluation image to each training image would include matching images that do not share a joint view of the scene, the matching candidates are preselected by the image retrieval approach mentioned in Section 7.3.2. Therewith, every evaluation image is compared to its ten nearest neighbours according to the image retrieval results. Figure 7.5 depicts results of an evaluation image – in the left column – matched to a training image from the (top), rendered (mid) and translated (bottom) dataset in the right column. Table 7.1 shows the average number of matches respectively inliers between the evaluation images and the images of the training datasets. Image-to-image translation on rendered images increases the number of inliers significantly from 12 to 79. The average number of

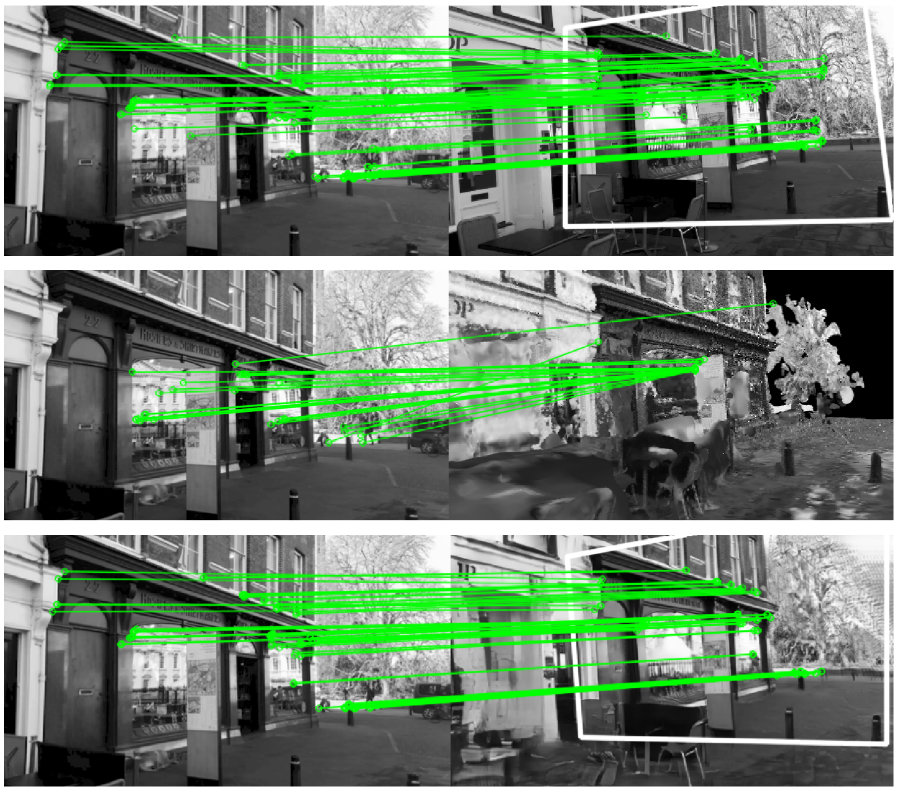


Figure 7.5: Exemplary visual results for feature matching. Each row shows matched features between a captured evaluation image (left) and a training image (right). The white bounding boxes depict the borders of the projected evaluation images. Training images are from top to bottom in the *captured*, *rendered* and *translated* domain. No image could be projected for the *rendered* domain, due to a lack of model computation from an insufficient number of inliers.

Dataset	Avg. # of matches	Avg. # of inliers	%
Captured	678	244	35.9
Rendered	240	12	5.0
Translated	396	79	19.9

Table 7.1: Average numbers of total matches and inliers between training images and evaluation images. The evaluation images are the captured images from the *Shop Façade* evaluation set. The last column shows the percentage of the average inliers in relation to the average matches.

12 inliers derived on the rendered images is generally not satisfying for computer vision tasks. However, after image-to-image translation the average number of inliers rises to 79, which is a decent amount of matches to successfully solve image registration or other image analysis tasks.

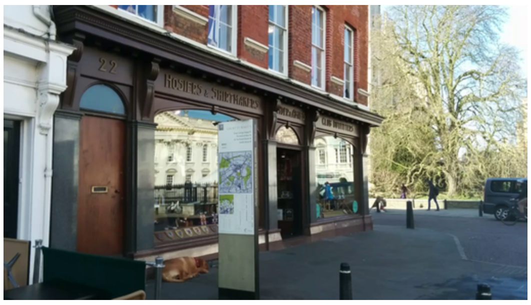
7.4.2 Image Retrieval

Image retrieval is evaluated on the captured training images, the rendered images and the translated images. The captured evaluation images serve for evaluation on all three training datasets. A visual representation of results is illustrated in Figure 7.6. Figure 7.6 (a) shows one of the evaluation images, whereas Figure 7.6 (b), (c) and (d) each show the four nearest neighbours of the training sets (*Captured*, *Rendered*, *Translated*) corresponding to the evaluation image. The mean pose differences are computed between each evaluation image and its top 1, top 4 and top 10 nearest neighbours for each training dataset (Table 7.2).

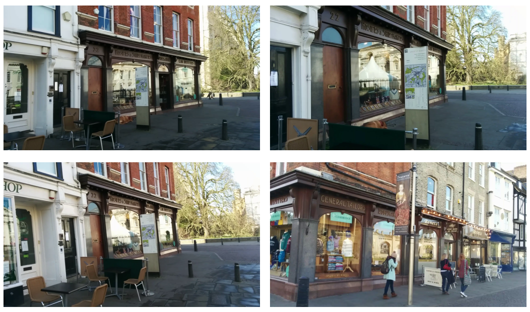
	Mean Pose Difference		
Dataset	Top 1	Top 4	Top 10
Captured	$0.72m, 0.43°$	$0.69m, 0.42°$	$0.82m, 0.50°$
Rendered	$2.62m, 0.84°$	$2.73m, 0.85°$	$2.88m, 0.89°$
Translated	$\mathbf{0.49m, 0.29°}$	$\mathbf{0.38m, 0.28°}$	$\mathbf{0.49m, 0.31°}$
Improvement	31.9%, 32.5%	44.9%, 33.3%	40.2%, 38.0%

Table 7.2: Mean pose differences of evaluation images to their nearest neighbours from the training datasets. The mean pose differences are computed between each evaluation image and its top 1, top 4 and top 10 nearest neighbours for each training dataset. Best results are highlighted in bold style.

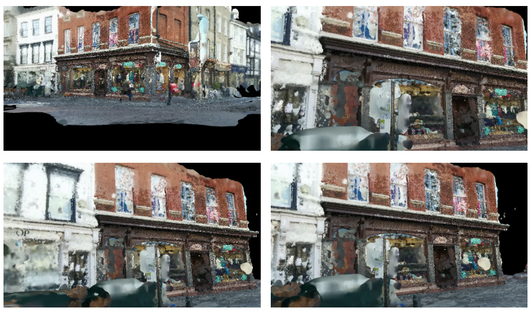
Utilizing translated images clearly leads to better results over the usage of merely captured images. Image retrieval benefits from the higher number of images leading to a denser sampling compared to captured images, hence finding images in the database closer to the query image and thus improving the pose estimate. Image retrieval trained on rendered images shows a decreased accuracy compared to retrieval on captured images due to high dissimilarity to the evaluation images.



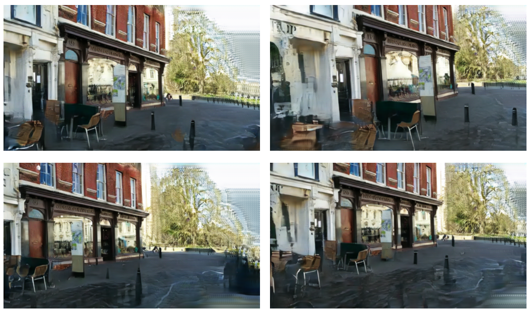
(a)



(b)



(c)



(d)

Figure 7.6: Exemplary results on image retrieval. (a) Shows an evaluation image in the *captured* domain. (b), (c) and (d) each show the four nearest neighbours to (a) in the *captured*, *rendered* and *translated* domain.

8 Experiments

This chapter presents the experiments on camera re-localization by data-driven methods under varying experimental setups. Experiments without DA are conducted in Section 8.1, experiments with DA by image rendering are conducted in Section 8.2 and experiments with DA by image-to-image translation are conducted in Section 8.3. Parts of this chapter have been published in the conference papers and journal article mentioned in the Chapters 5, 6 and 7.

8.1 Localization without Data Augmentation

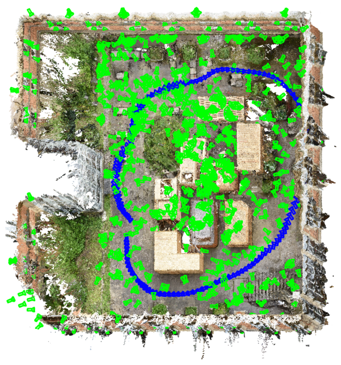
In this section, experiments on localization without DA are carried out. The focus is set on the evaluation of SqueezePoseNet which is compared to PoseNet and the modified VGG16-Net. The methodology covering these experiments is outlined in Chapter 5. The *medium* and *low coverage* evaluation sets in the atrium's scene introduced in Section 4.1 serve for the experiments. In Section 8.1.1 and 8.1.2 visual results as well as metric evaluations compared to ground truth are presented.

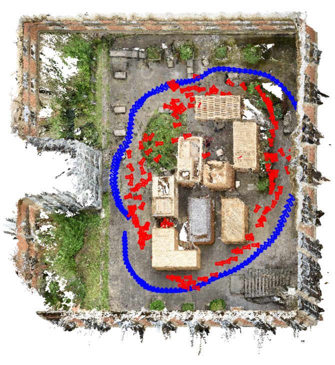
8.1.1 Medium coverage set

The *medium coverage* set shows a medium coverage of training and evaluation data in terms of similarity with respect to scene views as well as poses. Figure 8.1 (a) shows the training poses in green and the evaluation poses in blue for the *medium coverage* set. The estimated poses derived by the modified VGG16-Net and SqueezePoseNet are visualized in Figure 8.1 (b) respectively Figure 8.1 (c).

The histograms in Figure 8.2 show the derived errors to the ground truth poses. Figure 8.2 (a) and 8.2 (b) show the spatial errors of the modified VGG16-Net and SqueezePoseNet. Figure 8.2 (c) and 8.2 (d) show the related angular errors likewise.

The numerical results in Table 8.1 are represented by the median spatial and angular errors. The pose estimation accuracy is 4.91 m and 33.30° for the modified VGG16-Net, 5.19 m and 29.28° for SqueezePoseNet and 8.60 m and 50.83° for PoseNet. The table includes the numerical results of the *low coverage* set which is covered in the subsequent section.


(a)


(b)

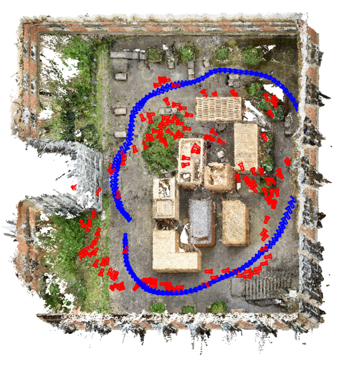
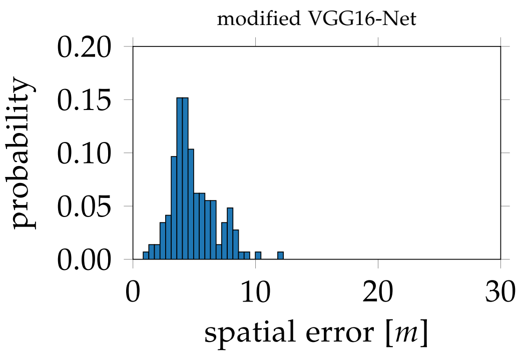
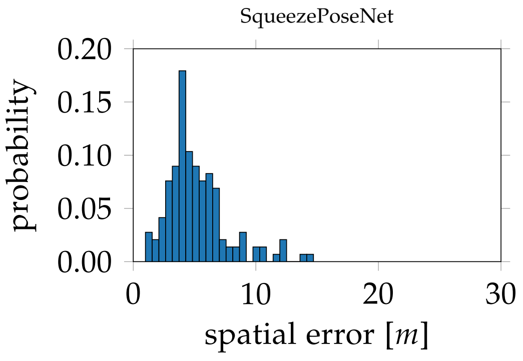

(c)

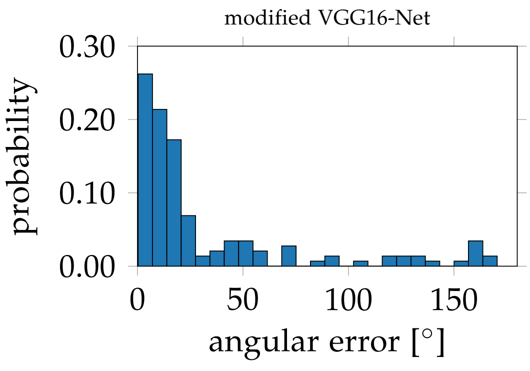
Figure 8.1: *Medium coverage* set. Training poses (green), evaluation poses (blue) and estimated poses (red). Visualization of (a) training poses and evaluation poses, (b) pose estimates derived by the modified VGG16-Net and (c) pose estimates derived by SqueezePoseNet.



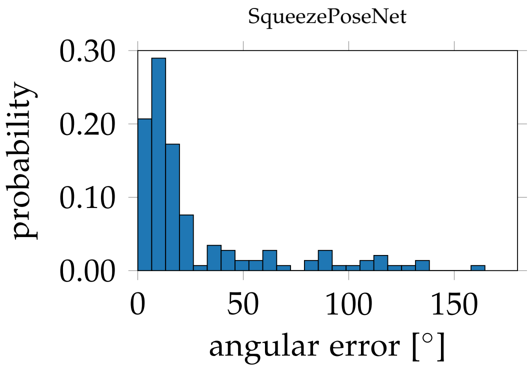
(a)



(b)



(c)



(d)

Figure 8.2: *Medium coverage* set. The histograms show the spatial and angular errors of the CNN-derived poses to ground truth. Whereas (a) and (b) depict the spatial errors of the modified VGG16-Net respectively SqueezePoseNet and (c) and (d) the angular errors likewise.

Dataset	PoesNet	VGG16-Net (mod.)	SqueezePoseNet
Medium coverage set	8.60 m, 50.83°	**4.91 m**, 33.30°	5.19 m, **29.28**°
Low coverage set	11.47 m, 46.40°	**11.34 m**, **37.33**°	15.18 m, 65.02°

Table 8.1: Evaluation errors on the *medium* and *high coverage* set. Bold text marks best result on a set.

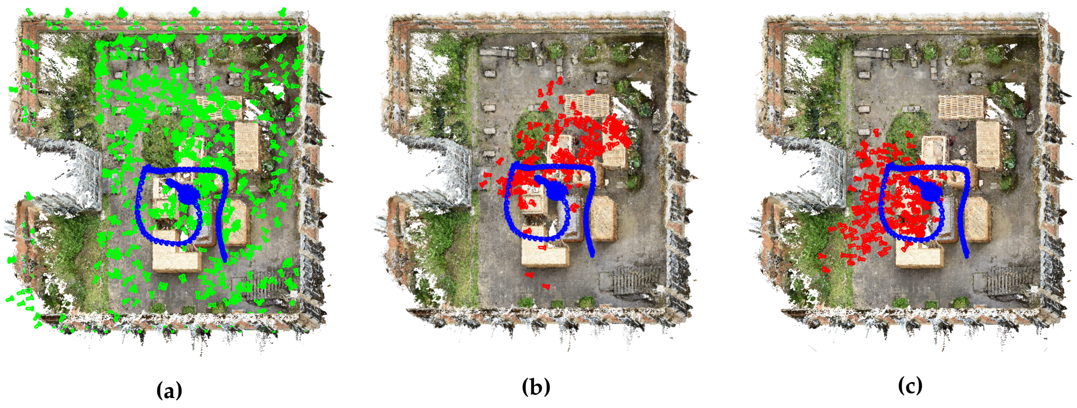


Figure 8.3: *Low coverage* set. Training poses (green), evaluation poses (blue) and estimated poses (red). Visualization of (a) training poses and evaluation poses, (b) pose estimates derived by the modified VGG16-Net and (c) pose estimates derived by SqueezePoseNet.

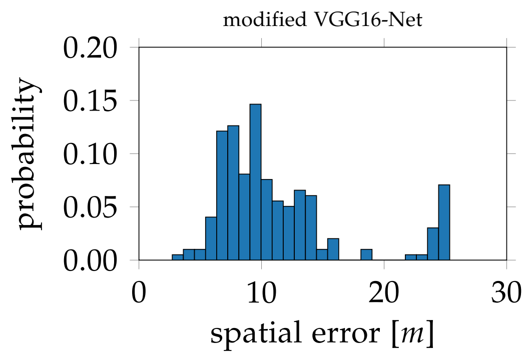
8.1.2 Low coverage set

This set shows a low coverage between training poses and evaluation poses. Figure 8.3 (a) shows the training poses in green and the evaluation poses in blue for the *low coverage* set. The estimated poses derived by the modified VGG16-Net and SqueezePoseNet are visualized in Figure 8.3 (b) respectively 8.3 (c).

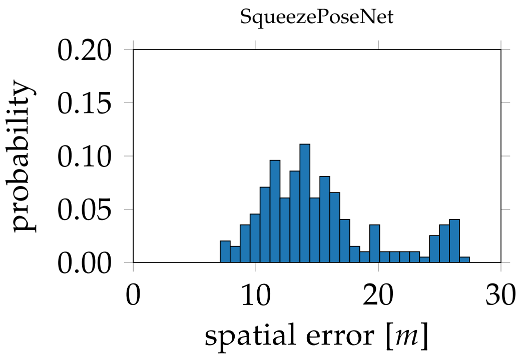
The histograms in Figure 8.4 show the derived errors to the ground truth poses. Figure 8.4 (a) and 8.4 (b) show the spatial errors of the modified VGG16-Net and SqueezePoseNet. Figure 8.4 (c) and 8.4 (d) show the related angular errors likewise.

The numerical results in Table 8.1 represent the spatial and angular errors. The pose estimation accuracy is 11.34 m and 37.33° for the modified VGG16-Net, 15.18 m and 65.02° for SqueezePoseNet and 11.47 m and 46.40° for PoseNet.

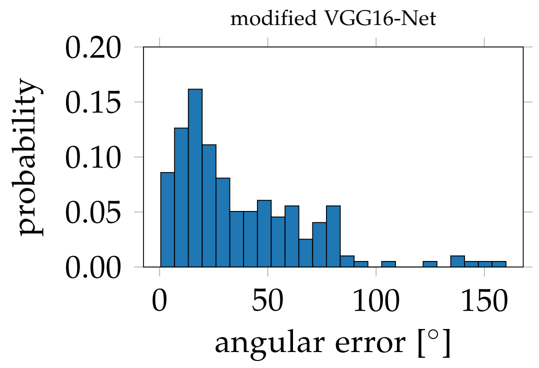
The experiments on both, the *medium* and the *low coverage* set, depict an improvement of the modified VGG16-Net over the shallower PoseNet (Table 8.1). The lightweight SqueezePoseNet scores confident results on the *medium coverage* set with similar performance as the modified VGG16-Net and even outperforming PoseNet while having a smaller model size than both of these networks. Experiments on the *low coverage* set depict unsatisfying results over all utilized CNNs. The pose errors assumedly arise due to the low similarity between training and evaluation data. Figure 8.5 shows an overview of the distribution of training poses (green), evaluation poses (blue) and estimated poses (red). It is clearly visible that the network is not capable of extrapolating from the training poses – which are mainly on the bottom of the scene – to the evaluation poses – which are at the top of the scene. The estimates lie in between and are not extrapolated to the full extent.



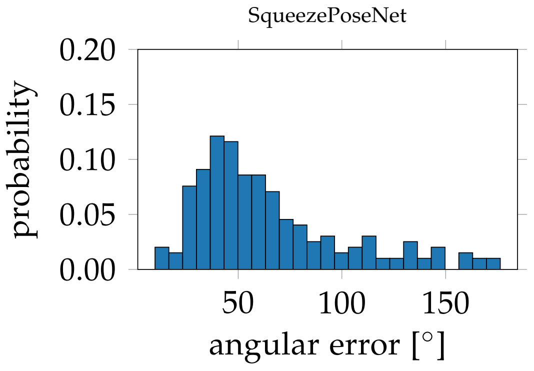
(a)



(b)



(c)



(d)

Figure 8.4: *Low coverage* set. The histograms show the spatial and angular errors of the CNN-derived poses to ground truth. Whereas (a) and (b) depict the spatial errors of the modified VGG16-Net respectively SqueezePoseNet and (c) and (d) the angular errors likewise.

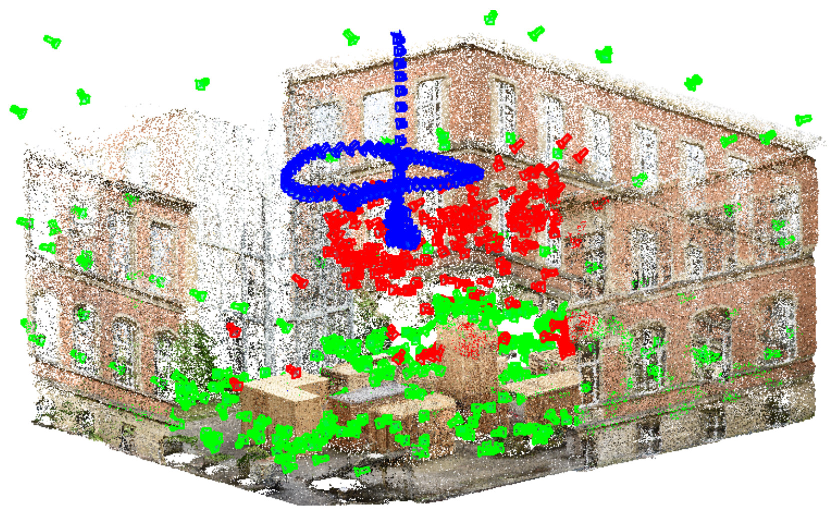


Figure 8.5: Camera poses of the training images (green), evaluation images (blue) and estimates (red) of the *low coverage* set. Is is clearly visible that the network is not capable of extrapolating from the training poses – which are mainly on the bottom of the scene – to the evaluation poses – which are at the top of the scene. The estimates lie in between, and are not extrapolated to the full extent.

8.2 Localization with Data Augmentation by Image Rendering

Concerning the localization results in Section 8.1 which are shown to be highly affected by the insufficient distribution of training data, image rendering is utilized to tackle this issue in this section. The feasibility of utilizing rendered images for DA to improve localization is initially evaluated on the *Puzzle* dataset and subsequently on the more complex *Atrium* dataset. A profound description of both datasets can be found in Section 4. The methodology covering these experiments is outlined in Chapter 6.

8.2.1 Puzzle

To tackle and overcome the issue of the experienced limited extrapolation capability of CNNs for camera re-localization due to a disadvantageous distribution of training and evaluation data, the *Puzzle* dataset is created and augmented with rendered images. The limitation considering extrapolation is in large parts caused by the training data, specifically by the lack of training data in the spatial neighborhood of the evaluation data. Therefore, a network is not able to learn, that poses appear in particular locations. By adding images of the scene in the spatial neighborhood of the evaluation data, the results could be enhanced. However, such images do generally not exist and it is expectable that on application level, such data is neither available. Therefore, images are rendered in the scene of the puzzle to augment the training dataset.

In these experiments, PoseNet and the modified VGG16-Net are trained and evaluated on the original *Puzzle* dataset and on the *Puzzle* dataset augmented with rendered images. The results on PoseNet and the modified VGG16-Net are visualized in Figure 8.6 respectively Figure 8.7.

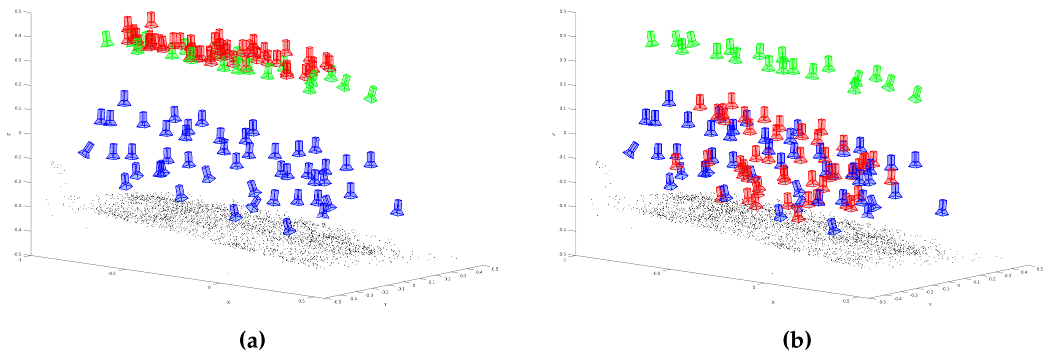


Figure 8.6: Visualization of pose results by PoseNet on the *Puzzle* dataset. (a) Pose results for PoseNet trained on the original *Puzzle* dataset. (b) Pose results trained on the augmented *Puzzle* dataset. Green cameras depict training poses of the original *Puzzle* dataset, blue cameras depict evaluation poses and red cameras depict estimated poses derived by PoseNet. The black points indicate the puzzle's point cloud. It is clearly shown, that the rendered images in the augmented dataset support the extrapolation potential of the network.

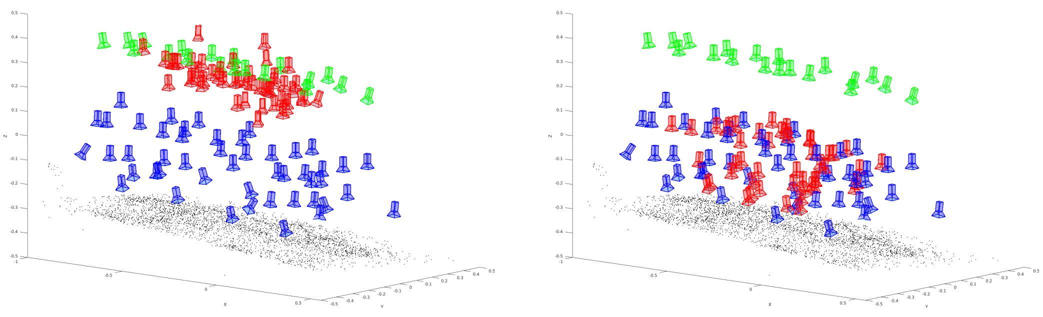


Figure 8.7: Visualization of pose results by the modified VGG16-Net on the *Puzzle* dataset. (a) Pose results for the modified VGG16-Net trained on the original *Puzzle* dataset. (b) Pose results for the modified VGG16-Net trained on the augmented *Puzzle* dataset. Green cameras depict training data of the original *Puzzle* dataset, blue cameras depict the ground truth and red cameras depict estimated poses derived by the modified VGG16-Net. The black points indicate the puzzle's point cloud. It is clearly shown, that the rendered images in the augmented dataset support the extrapolation potential of the network.

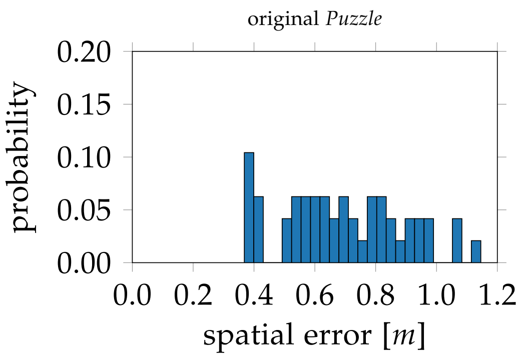
The numerical results for PoseNet and the modified VGG16-Net are shown in Table 8.2. For PoseNet trained on the original *Puzzle* dataset, the evaluation errors are 0.47 m and 16.07° for the spatial respectively angular component. The evaluation errors on PoseNet, trained on the augmented dataset with rendered images are 0.18 m and 7.53°. This is an improvement of 61.7% for the spatial component and 53.1% for the angular component. In Figure 8.8 the evaluation errors of PoseNet are visualized in histograms.

Dataset	PoseNet	modified VGG16-Net
original *Puzzle*	0.47 m, 16.07°	0.39 m, 17.07°
augmented *Puzzle*	**0.18 m**, **7.53**°	**0.16 m**, **4.10**°
Improvement	61.7%, 53.1%	59.0%, 76.0%

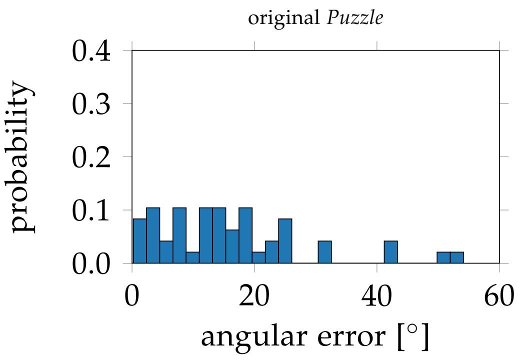
Table 8.2: Evaluation errors for PoseNet and the modified VGG16-Net trained on the original *Puzzle* dataset and the augmented *Puzzle* dataset. The bottom column shows the percental improvement of training on the augmented dataset against the original. The best results for each CNN is marked bold.

For the modified VGG16-Net trained on the original *Puzzle* dataset, the evaluation errors are 0.39 m and 17.07° for the spatial respectively angular component. The evaluation errors on the modified VGG16-Net, trained on the augmented dataset with rendered images are 0.16 m and 4.10°. This is an improvement of 59.0% for the spatial component and 76.0% for the angular component. In Figure 8.9 the evaluation errors of the modified VGG16-Net are visualized in histograms.

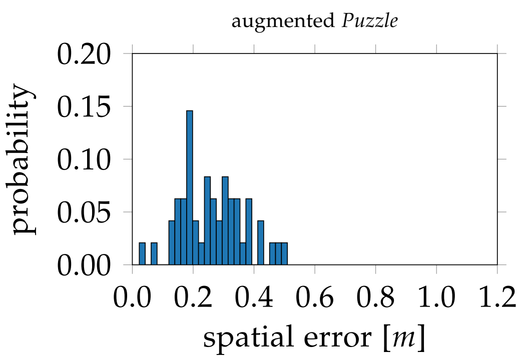
It is clearly visible from the numerical results in Table 8.2 and from the histograms in Figure 8.8 and 8.9, that DA by image rendering achieves better localization results than localization without DA. This is confirmed by the visual representations in Figure 8.6 and 8.7.



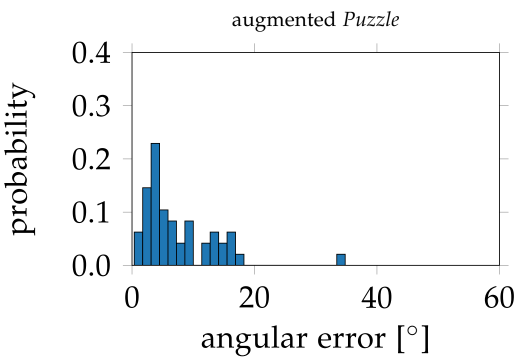
(a)



(b)



(c)



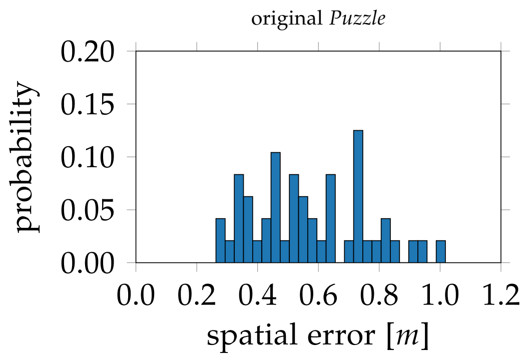
(d)

Figure 8.8: The histograms show the spatial and angular pose errors derived by PoseNet. Histograms (a) and (c) depict the spatial respectively angular errors based on training on the original *Puzzle* dataset. Histograms (b) and (d) depict the spatial and angular errors based on training on the *Puzzle* dataset augmented with rendered images. The errors could be decreased from 0.47 *m* and 16.07° to 0.18 *m* and 7.53° by the augmentation.

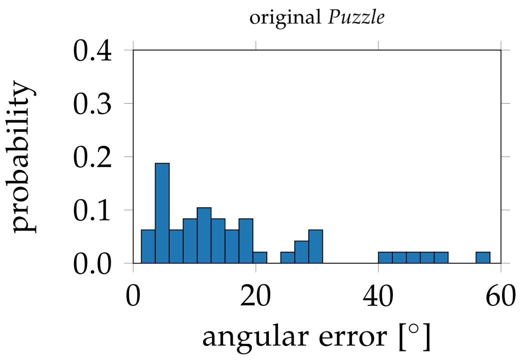
8.2.2 Atrium

Analogous to the experiments in Section 8.2.1, DA by image rendering is carried out on the *Atrium* dataset. In contrast to the 3D model of the puzzle, where only a single textured plane forms the scene, the atrium represents a more complex scene consisting of multiple geometric objects and diverse textures. Experiments are carried out with the modified VGG16-Net and SqueezePoseNet. Both networks are trained on the different training datasets *Captured*, *Coincide*, *Captured+Coincide*, *Diverge* and *Captured+Diverge* as introduced in Section 6. The evaluation is carried out on the *medium* and *low coverage* set. The results on the *medium* and *low coverage* set are depicted in Table 8.3 respectively Table 8.4. A visual representation of the evaluation errors separated by the utilized CNNs is depicted in Figure 8.10. The figure depicts the position and orientation errors separated by the *medium* and *low coverage* set.

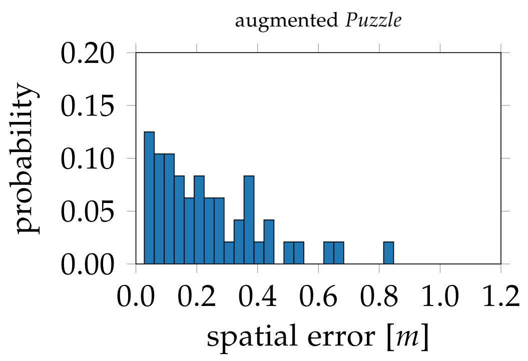
An illustration of the best results for the *medium coverage* set and the *low coverage* set is depicted in Figure 8.11. The single figures show estimated poses on the *Captured* dataset (a), (c), (e), (g) as well as the estimated poses on the respective augmented datasets scoring



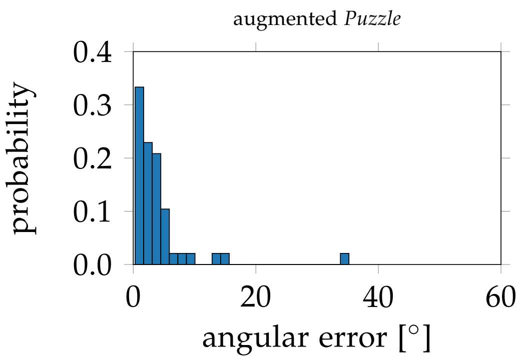
(a)



(b)



(c)



(d)

Figure 8.9: The histograms show the spatial and angular pose errors derived by VGG16-Net. Histograms (a) and (c) depict the spatial respectively angular errors on the original *Puzzle* dataset. Histograms (b) and (d) depict the spatial and angular errors on the *Puzzle* dataset augmented with rendered images. The errors could be decreased from 0.39 *m* and 17.07° to 0.16 *mand*4.10° by the augmentation.

Dataset	modified VGG16-Net	SqueezePoseNet
Captured	4.91 *m*, 33.30°	5.19 *m*, 29.28°
Coincide	3.36 *m*, 21.83°	5.18 *m*, 27.45°
Captured+Coincide	3.37 *m*, 19.63°	**3.91 m, 19.01**°
Diverge	3.90 *m*, 21.79°	5.32 *m*, 25.99°
Captured+Diverge	**3.14 m, 18.40**°	3.89 *m*, 19.90°
Improvement	36.05%, 44.74%	24.66%, 35.08%

Table 8.3: Evaluation errors on the *medium coverage* set. The improvement corresponds to the best result per method, which is marked in bold.

Dataset	modified VGG16-Net	SqueezePoseNet
Captured	11.34 m, 37.33°	15.18 m, 65.02°
Coincide	**4.53 m, 16.67°**	5.26 m, 24.90°
Captured+Coincide	4.46 m, 20.90°	6.60 m, 31.88°
Diverge	4.48 m, 26.76°	**4.65 m, 24.96°**
Captured+Diverge	6.38 m, 19.02°	6.86 m, 26.43°
Improvement	60.05%, 55.34%	69.37%, 61.61%

Table 8.4: Evaluation errors on the *low coverage* set. The improvement corresponds to the best result per method, which is marked in bold.

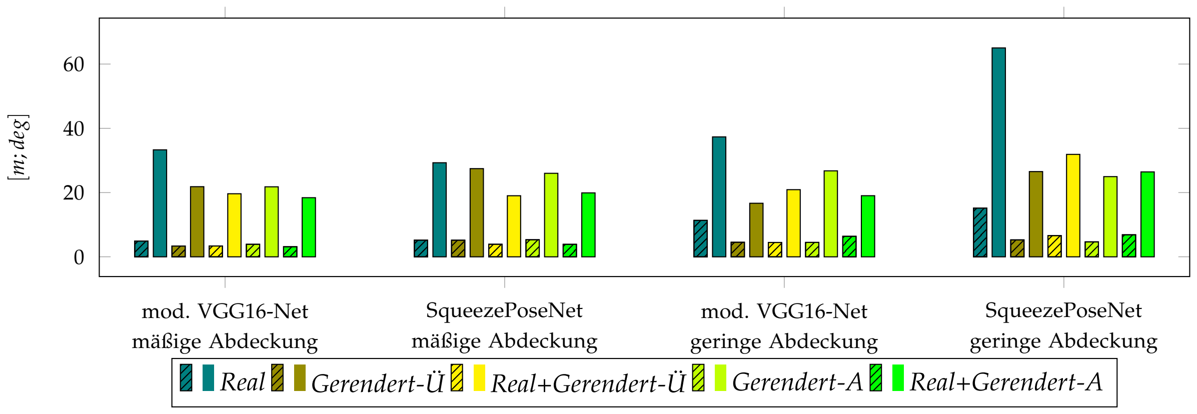


Figure 8.10: Visualization of evaluation errors. The Figure depicts the position errors (striped bars) and orientation errors (untextured bars) errors separated by the *medium* and *low coverage* set evaluated on the modified VGG16-Net and SqueezePoseNet. The experiments are carried out on the training datasets types *Captured*, *Coincide*, *Captured+Coincide*, *Diverge* and *Captured+Diverge*.

best results (b), (d), (f), (h). The best results are scored by the following networks: (a) Modified VGG16-Net trained on the *Captured* dataset. (b) Modified VGG16-Net trained on the *Captured+Diverge* dataset. (c) SqueezePoseNet trained on the *Captured* dataset. (d) SqueezePoseNet trained on the *Captured+Coincide* dataset. (e) Modified VGG16-Net trained on the *Captured* dataset. (f) Modified VGG16-Net trained on the *Coincide* dataset. (g) SqueezePoseNet trained on the *Captured* dataset. (h) SqueezePoseNet trained on the *Diverge* dataset.

The experiments show a clear improvement of utilizing DA by image rendering. The rendered images are used to extend the distribution of scene views and poses for the training of CNNs. The accuracy is improved when utilizing DA for all experiments.

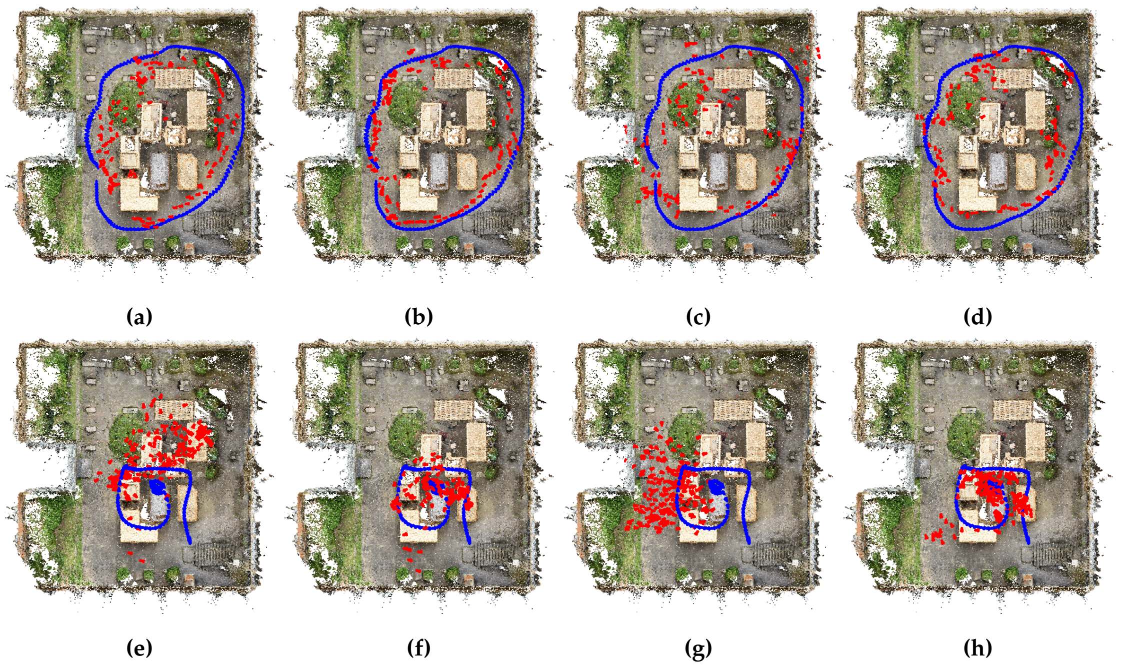


Figure 8.11: Visualization of evaluation poses (blue) and estimated poses (red) for the *medium coverage* set in the top row and the *low coverage* set in the bottom row. The single figures show estimated poses on the *Captured* dataset (a), (c), (e), (g) as well as the estimated poses on the respective augmented datasets scoring best results (b), (d), (f), (h). (a) Modified VGG16-Net trained on the *Captured* dataset. (b) Modified VGG16-Net trained on the *Captured+Diverge* dataset. (c) SqueezePoseNet trained on the *Captured* dataset. (d) SqueezePoseNet trained on the *Captured+Coincide* dataset. (e) Modified VGG16-Net trained on the *Captured* dataset. (f) Modified VGG16-Net trained on the *Coincide* dataset. (g) SqueezePoseNet trained on the *Captured* dataset. (h) SqueezePoseNet trained on the *Diverge* dataset.

8.3 Localization with Data Augmentation by Image-to-Image Translation

In this section, experiments on localization with DA by image-to-image translation are carried out. Experiments are carried out on the *Shop Façade* dataset and the *Atrium* dataset. Both datasets are introduced in Chapter 4. Image-to-image translation on the *Atrium* dataset leads to unsatisfying results. Experiments on camera re-localization are therefore not carried out on this dataset. Reasons on this are discussed in Chapter 9 nevertheless. The localization pipeline utilized for the experiments on the *Shop Façade* dataset is based on DSAC++ [Bra+17]. The image-to-image translation and the localization methods described in Chapter 7 are utilized for the subsequent experiments.

8.3.1 Shop Façade

For the evaluation of image-to-image translation, the localization approach presented in Chapter 7 is adapted. The datasets of the captured, rendered and translated images of the *Shop Façade* scene serve for training the network. The network is trained on each of the mentioned training sets separately. Additionally, a training of a joint training dataset containing the captured images and the translated images is realized. The evaluation of the networks is carried out on the evaluation dataset with the captured images on all four trained models, which are *Captured*, *Rendered*, *Translated* and *Captured+Translated*. All experiments are processed with the same settings of hyperparameters to maintain comparability. The results are depicted in Table 8.5. The network achieves a pose accuracy as median translation and rotation errors of $0.14\ m/0.7°$ on the captured data, $8.86\ m/39.5°$ on the rendered data and $0.16\ m/0.6°$ on the translated data. Training on the joint dataset of captured and translated images scores $0.12\ m$ and $0.4°$, which depicts the best result. Training on rendered images leads to unsatisfying results. Training on merely translated images scores similar results as training on the captured images, which is promising.

Dataset	Total Number of Images	Median Pose Error
Captured	231	$0.14\ m, 0.7°$
Rendered	2652	$8.86\ m, 39.5°$
Translated	2652	$0.16\ m, 0.6°$
Captured+Translated	$231 + 2652$	$\mathbf{0.12\ m, 0.4°}$

Table 8.5: Median pose errors on the *Captured*, *Rendered*, *Translated* and *Captured+Translated* datasets. The joint *Captured+Translated* dataset scores the best results. Training on rendered images leads to unsatisfying results. Training on merely translated images scores similar results as training on the captured images, which is promising.

9 Discussion

In this chapter, the results of the experiments are concluded and discussed. The chapter is subdivided into discussions according to localization without DA, localization with DA by image rendering and localization with DA by image-to-image translation.

9.1 Localization without Data Augmentation

The experiments on localization without DA in Section 8.1 show that CNNs are generally capable to estimate image poses but are limited in terms of extrapolation. That means, the bare space of training poses limits the performance of such networks profoundly. The evaluation is carried out on several CNNs, including SqueezePoseNet which has a small model size and is therefore well suited for mobile applications like small robots, MAVs or mobile handheld devices. The approach on building a small network is evaluated on the *medium* and the *low coverage* set of the *Atrium* dataset and compared to deeper networks, like PoseNet and the modified VGG16-Net. The modified VGG16-Net as well as SqueezePoseNet score similar results on the *medium coverage* set, whereas the modified VGG16-Net has a deeper network structure, therefore more parameters and a bigger model size. Both networks, with a position estimation error of 4.91 m respectively 5.19 m, could provide initial poses for a further and optional pose refinement step or generally could serve for pose initialization for subsequent processes. The estimation of poses with SqueezePoseNet is processed in less than 5 ms, representing real-time capability for almost all applications.

Less accurate results are obtained on the *low coverage* set, where hardly any similarity to the evaluation data is provided in the training dataset. The spatial differences of 11.34 m and 15.18 m in a scene of about $39 \times 36 \times 18$ m^3 does not satisfy the needs on camera re-localization. Also the orientation estimates between 37.33° and 65.02° are deficient to serve for further processing. The majority of the position error is contributed by a spurious estimation of the height component. The poses are systematically estimated spatially too low. It is reasonable that the CNN can not extend its knowledge too far from the trained data, as it has never learned such extends from the training data. Even though interpolation works well, the extrapolation on the other hand performs less accurate. It can be stated, that a high coverage of training data serves better to train CNNs for pose regression than a low coverage. The higher the similarity and overlapping views between training and testing images is, the better are the pose estimates.

To overcome the limitation of dissimilar training and evaluation data, solutions utilizing DA are presented in Chapter 6. Since it is unpractical to collect such missing data of the scene with a high density, additional views are rendered to augment the original training dataset and to achieve such density.

9.2 Localization with Data Augmentation by Image Rendering

The general performance of CNNs for localization can potentially be improved by providing a higher distribution of training data. This can be achieved with DA by image rendering. The methodology of this process is outlined in Chapter 6 and the associated experiments are provided in Section 8.2. In this section, the results of the experiments are outlined and discussed.

The results of the training processes enhanced by DA show thoroughly positive outcome. The experiments on the *Puzzle* dataset depict the benefits of DA by image rendering as the training space of poses and images is enhanced leading to accuracy improvements of about 60% in position and up to 70% in orientation. The rendered images from the *Puzzle* dataset are generated with a high realistically appearance as a photo-realistic 3D model could be provided. Since such models are seldom provided, a 3D model with lower quality is used for subsequent experiments. Therefore, the *Atrium* dataset serves for further experiments.

The accuracy of pose estimation is increased for the modified VGG16-Net and SqueezePoseNet on both evaluation datasets of the *Atrium* dataset, the *medium coverage* set and the *low coverage* set. The improvements for the *medium coverage* set are up to 36.05% for the translation component and up to 44.74% for the orientation component. The improvements for the *low coverage* set are up to 69.37% for the translation component and up to 61.61% for the orientation component. From the numerical and visual presentations of the results, it can be seen that the pose estimates move closer the the ground truth when adding rendered images to the training process. However, the knowledge transfer from captured images to rendered images is merely moderate by the CNN due to the dissimilarity of the image domains. Considering the *medium coverage* set, this can be stated since the *Coincide* dataset, which shares the exact poses as the evaluation images does improve the accuracy, but still scores not remarkable results. The same applies for the *Diverge* dataset, which also contains solely rendered images. However, a combination of captured and rendered images improves the accuracy significantly, leading to clearly better results than training on captured images only. Considering the *low coverage* set, reviewing the numerical results shows also a clear improvement of localization with DA. Whereas training on the *Captured* dataset scores insufficient results, the training on the proposed datasets enhanced by DA improved the pose estimation. However, by visualizing the pose estimates in Figure 8.11 (e) – (h) it is shown that the poses are still not determined satisfactorily. The numerical improvement is mainly caused by the fact of better distributed training data, hence the estimated image poses are shifted more towards the actual evaluation poses.

9.3 Localization with Data Augmentation by Image-to-Image Translation

In the experiments concerning DA by image-to-image translation (Section 8.3), images in the *rendered* domain are translated to images in the *captured* domain. The methodology of this procedure is outlined in Chapter 7. With the experiments on translated images, enhancements over the usage of rendered images are shown for localization.

DA by image-to-image translation shows beneficial impact compared to training on merely captured data. The utilized localization network scores higher accuracies training on a joint dataset of captured and translated images than on merely captured or rendered images. Training on only translated images scores similar results as training on captured images. The network trained on rendered images achieves results that are insufficient. That implies that the network potentially learns representations for rendered images, which can not be transferred to the captured images for evaluation. Overall, it is shown that image-to-image translation can translate rendered images into valuable training data. Compared to captured images, the experiments on translated images also show improvements regarding image retrieval (Section 7.4).

In the following a few failure cases of image-to-image translation are shown. In contrast to the experiments on the *Shop Façade* dataset, tests are also carried out on more complex scenes. These tests are carried out on the introduced *Atrium* dataset and the *Dancing House* dataset [Sat+18b]. Exemplary results of the image-to-image translation are illustrated in the Figures 9.1, 9.2 and 9.3. Rendered images are each depicted on the left and their corresponding translated images on the right. Figure 9.1 depicts two images pairs of unsatisfying translations in the atrium scene. In both examples, several windows are rotated about 90°. Assumedly, high level features describing a window trigger at wrong locations, which could be caused by scale or orientation variances within the network since windows are present in different scales and orientations over the training samples. Figure 9.2 depicts example images of more satisfying translations in the atrium scene. The windows in image (b) appear more realistic than in the previous example. The example in image (d) looks promising, showing that image-to-image translation can generate satisfactory output for that scene. The images in Figure 9.3 show the *Dancing House* scene with decent results illustrated in Figure 9.3 (b). Figure 9.3 (c) shows a rendered image generated far away from the original space of poses given by the training data. Within the image-to-image translation pipeline, no image with a similar pose is provided for that view. The translated image in Figure 9.3 (d) is generated deficient and can not be considered for subsequent processing. The network is not able to generate a realistic image from a view, where no training samples with similar views exist nearby. The extrapolation capability is therefore limited.

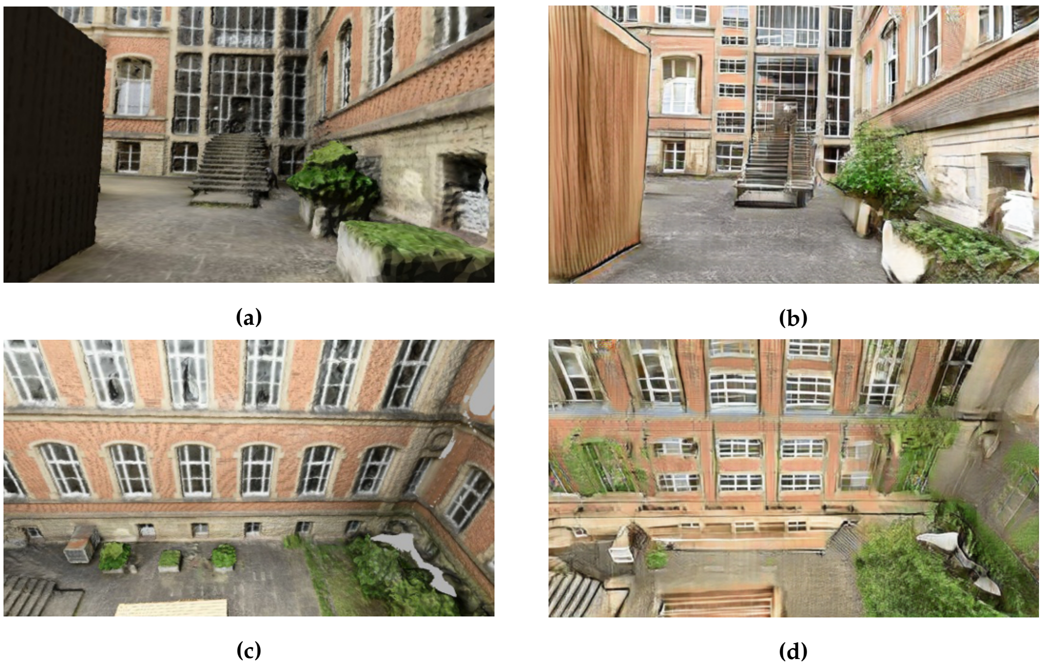


Figure 9.1: Negative examples of image-to-image translation. Input images in the *rendered* domain are on the left and corresponding translated images in the *captured* domain are on the right. In both examples, it seems that high-level features like windows are firing at the position of the windows joists. Assumedly, they are activated in a lower-scale level as they should be.

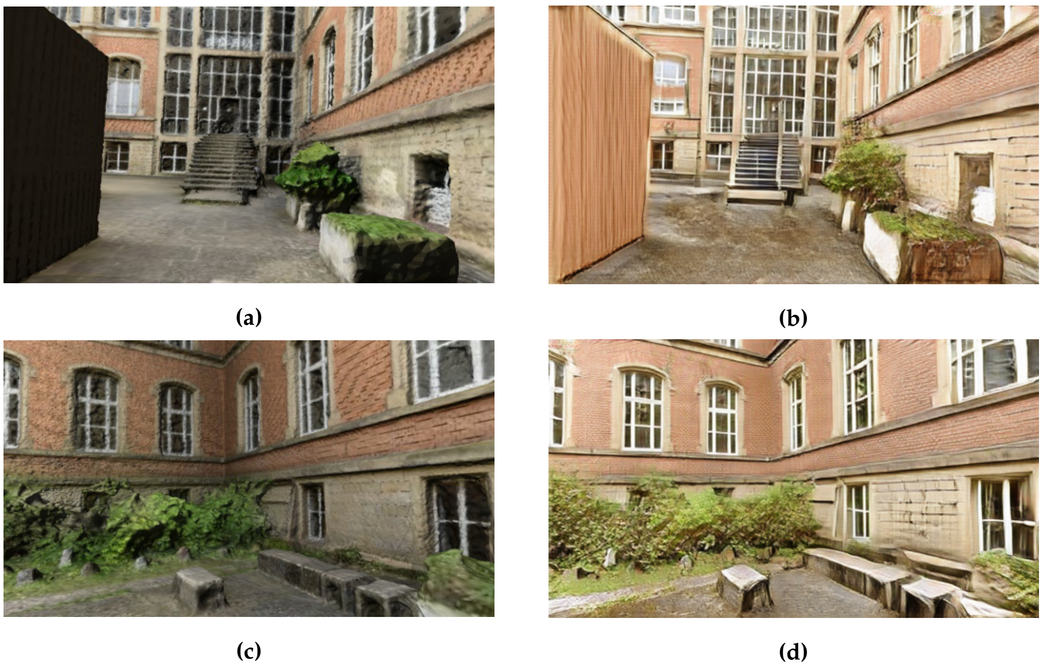


Figure 9.2: Positive examples of image-to-image translation. Input images in the *rendered* domain are on the left and corresponding translated images in the *captured* domain are on the right. Both examples depict satisfying results and an improvement regarding visual appearance of the scene. The windows in the back of image (b) are depicted correct, whereas the same windows in Figure 9.2 (b) are depicted false.

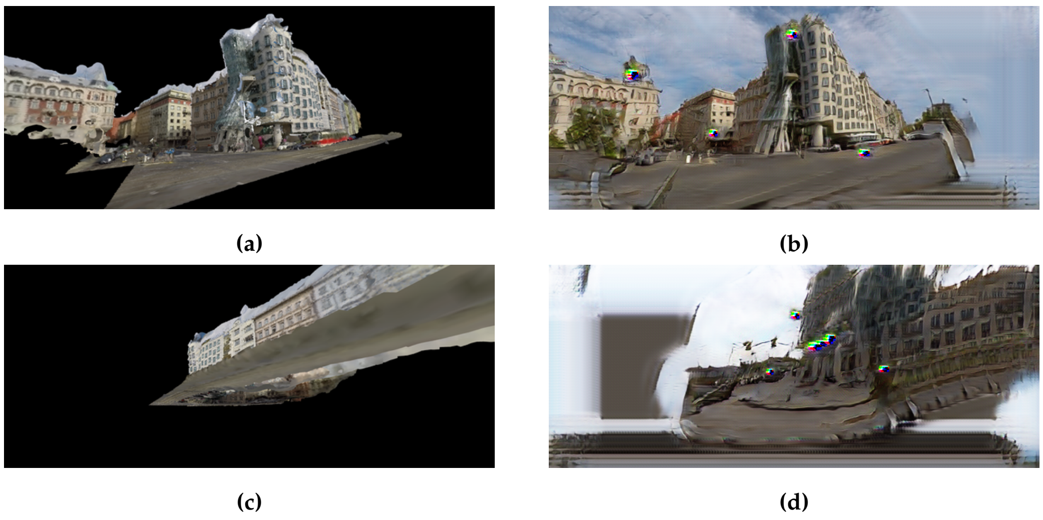


Figure 9.3: Negative examples of image-to-image translation. Input images in the *rendered* domain are on the left and corresponding translated images in the *captured* domain are on the right. Image (b) shows decent results being mapped from image (a). Image (c) shows an image rendered far away from any original training pose. Within the image-to-image translation pipeline, no image with a similar view is therefore provided. The translated image in (d) is generated deficient and can not be considered for subsequent processing.

10 Conclusion and Outlook

This chapter summarizes and concludes the results and contributions of the preceding chapters in Section 10.1 and gives an outlook in Section 10.2 as well as insights of potential future work.

10.1 Conclusion

CNN-based solutions for camera re-localization can satisfy the needs of estimating image poses within a few milliseconds. The accuracy and robustness of approaches based on end-to-end learning merely are less accurate than hand-crafted or hybrid localization approaches combining well-known models with end-to-end learning. End-to-end approaches, like PoseNet, SqueezePoseNet or image-based localization using LSTMs can serve to estimate coarse poses in little time to initialize navigation frameworks or for subsequent pose refinement. Besides, SqueezePoseNet is a lightweight framework with a small number of parameters and therefore has a small model size, being advantageous considering mobile platforms with low hardware capacity.

A drawback of the data-driven approaches is the necessity of providing adequate training data. A pose can only be determined if sufficient training data of the scene is available. If no sufficient training data is given or when the training should generally be improved with additional data, DA is a valuable process and covered in the Chapters 6 and 7. These chapters introduce localization with DA by image rendering respectively DA by image-to-image translation. Image rendering allows to generate training data with a higher distribution of scene views and image poses. This enriches training data and leads to improved performance of networks trained on such enhanced data. Image rendering has shown to enhance the localization thoroughly with improvements of up to 69% for translation and up to 61% for rotation in a complex scene. Further, DA by image-to-image translation has improved the localization by mapping images from a *rendered* domain into images in a *captured* domain on a benchmark dataset, where image rendering failed due to low quality rendering. Therewith, a hybrid network for camera localization is trained merely on synthetic data (translated images) and achieves results in the same quality compared to training on manually captured data. The accuracy of camera re-localization is improved by the supported training with translated images. Camera re-localization tasks with challenging training data will likely benefit even more from image-to-image translation, e.g. training data with a high spatial dissimilarity between training and evaluation images respectively higher discrepancies between their poses. Furthermore, image-to-image translation clearly improves the performance on image retrieval tasks, profiting from the higher distribution of training poses. Quality assessment on feature matching shows also an improvement utilizing translated images over rendered images.

10.2 Outlook

In this thesis, a workflow of improving data-driven learning tasks for camera re-localization is presented by DA from reconstructed 3D models. These 3D models are especially reconstructed only by the provided training data necessary for the re-localization task. A general drawback of the re-localization problem is this necessity of captured images that have to be provided or pre-captured at any time. However, image data is captured all over the globe and commercially provided by companies or publicly available shared by private persons. A well known example is the reconstruction of parts of the city of Rome by using merely publicly available online data [Aga+11]. Furthermore, there is satellite and aerial imagery available covering urban and rural areas. Such data can serve easily for training data for camera re-localization. Likewise to image data, 3D models of the entire earth's surface and its objects – partly indoors – are existing. Such 3D models are increasingly generated over the last decade and refined concerning their quality constantly. Such models are of interest for methods of autonomous driving, AR and general navigation or path planning. Tasks, that are addressed in computer vision, photogrammetry, robotics or geo-information sciences. The development of such 3D models is therefore of high interest for a wide range of users. That way 3D models serve for localization. Simultaneously, applications of visual localization may in return provide the most recent data to update such 3D models continuously while operating in-situ. This is therefore a win-win situation, as the models provide information for a localization and in return the localization applications provide captured image data that serves to update the 3D models. Therefore, the 3D model is kept prevailed up-to-date which in turn improves the localization. Considering high performance data transfer, this interaction could probably be provided in real-time. A linked network of cars, UAVs and/or pedestrians could therefore support each other with valuable information.

Of further interest is the amount of data content which can generally be learned by CNNs, especially by small CNNs. Even though the model size of CNNs does not increase with a rising amount of training images, a CNN should only be able to recognize a finite part of the training data. Small CNNs like SqueezePoseNet may fail or at least lose accuracy with an increase of the spatial dimension of a training scene. Investigations on network sizes and their capability to effectively learn from large distributions of data is of interest for future work.

The hybrid camera re-localization approach DSAC++, that combines a CNN followed by a spatial estimation method reaches state-of-the-art accuracies and outperforms methods based on hand-crafted features on several indoor and small-scale outdoor benchmark datasets. However, on large-scale datasets feature-based localization methods like AS score higher accuracies. This implies, that the hybrid network may not recognize the training data in its full extent. Considering a large or ambiguous scene, single networks may not cover the domain well. Therefore Mixture of Experts [Jac+91] is used to improve localization [BR19]. Multiple networks are trained on a local part of the scene to solve these ambiguities. A gating network chooses which of the expert models is responsible for a given input during runtime. It is of further interest to tackle the issue of limited capacity of CNNs considering the quantity of training data.

Many problems that have been successfully tackled with data-driven learning have in common, that the mapping from the input to the output is difficult to characterize by distinct mathematical models. It is suggested that mathematical models, that are well known to the user, should be implemented with respect to that knowledge in a hand-crafted manner. Exceptions are mathematical models that are computationally too expensive for real-time applications and be better substituted with learned algorithms. An example are the Navier-Stokes equations, describing the movement of fluid particles. Although the mathematical model is known, it is too complex to be solved in real-time manner for common applications and is substituted with a Random Forest approach being faster and more stable than numerical approaches [Lad+17]. Any mappings between an input and output that can not be described thoroughly with hand-crafted mathematical models should be considered to be learned in a data-driven manner. Considering camera re-localization, the generation of suitable features is challenging considering variational input. Finding suitable features under changing scene conditions like day and night is hard to achieve by hand-crafted methods. Such tasks should be tackled in a data-driven manner. On the other hand, the computation of camera poses from corresponding features, like 2D-3D correspondences, is well known and should be designed in a hand-crafted way. Therefore, the methodological focus should be set on the development of hybrid methods for localization. Processes in the pipeline that are well known or can be precisely described with mathematical models should be designed by hand. Processes that are not well known or unknown should be learned from data. This statement applies not only on camera re-localization, but in general.

Future work should further focus on the generalization of CNNs. Humans train their neural system from the date of birth. By time, a human can solve problems in different fields with satisfying robustness and accuracy. New problems can be approached fast by transfer learning with the knowledge from previous solved tasks. Even though, inspired by the human brain, CNNs are not identical to it. However, compared to humans, CNNs are trained a narrow with amount of data and often only for one particular and specific task. It is shown that transfer learning on CNNs improves the speed of convergence, the robustness and the accuracy. For the future, it will be interesting to trace two ways. One may focus on developing deep learning architectures that can be trained on a little amount of data – like zero-, one-, or few-shot learning [FFP06] and still achieve satisfactory outcome. The other is depending on hardware and software development and is to create frameworks that can be efficiently trained on enormous amounts of data. Up to now training on such enormous amounts is time-consuming, while the training itself is a bottleneck for fast research.

Considering image-to-image translation for DA, it is shown in this thesis to improve camera re-localization. In terms of utilizing 3D models for renderings, it is valuable to map such rendered images to a more realistic domain that shares the characteristics of potential evaluation respectively application data. Bigger gains are possible when translating rendered images from views that are substantially different from the captured views. However, generating plausible mappings for such views is harder, creating the necessity for further research to handle large pose changes between captured and rendered images. Also considering style transfer to extend the distribution of characteristics in the training data may serve for better generalization to unseen data and is therefore proposed for future work.

Bibliography

[Aga+11] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski: *Building Rome in a Day*. Vol. 54. 10. 2011, pp. 105–112 (cit. on p. 94).

[Agi17] Agisoft: *Agisoft LLC*. 2017. URL: `http://www.agisoft.com/` (visited on 10/09/2019) (cit. on pp. 37, 54).

[ABD12] P. F. Alcantarilla, A. Bartoli, and A. J. Davison: *KAZE Features*. *European Conference on Computer Vision*. 2012, pp. 214–227 (cit. on p. 16).

[Alt+16] H. Altwaijry, E. Trulls, J. Hays, P. Fua, and S. Belongie: *Learning to Match Aerial Images with Deep Attentive Architectures*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 3539–3547 (cit. on p. 43).

[Ano+18] A. Anoosheh, E. Agustsson, R. Timofte, and L. Van Gool: *Combogan: Unrestrained Scalability for Image Domain Translation*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018, pp. 783–790 (cit. on pp. 14, 35, 64).

[Ano+19] A. Anoosheh, T. Sattler, R. Timofte, M. Pollefeys, and L. Van Gool: *Night-to-Day Image Translation for Retrieval-Based Localization*. *International Conference on Robotics and Automation*. 2019, pp. 5958–5964 (cit. on pp. 14, 35, 64, 66).

[AZ13] R. Arandjelovic and A. Zisserman: *All About VLAD*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 1578–1585 (cit. on pp. 5, 52).

[ARS14] M. Aubry, B. C. Russell, and J. Sivic: *Painting-to-3D Model Alignment via Discriminative Visual Elements*. *ACM Transactions on Graphics*. Vol. 33. 2. 2014, p. 14 (cit. on p. 64).

[BL15] A. Babenko and V. Lempitsky: *Aggregating Deep Convolutional Features for Image Retrieval*. *arXiv preprint arXiv:1510.07493*. 2015 (cit. on p. 52).

[BTV06] H. Bay, T. Tuytelaars, and L. Van Gool: *SURF: Speeded-Up Robust Features*. *European Conference on Computer Vision*. 2006, pp. 404–417 (cit. on p. 5).

[BTZ96] P. Beardsley, P. Torr, and A. Zisserman: *3D Model Acquisition from Extended Image Sequences*. *European Conference on Computer Vision*. 1996, pp. 683–695 (cit. on p. 20).

[BZM06] A. Bosch, A. Zisserman, and X. Muñoz: *Scene Classification via pLSA*. *European Conference on Computer Vision*. 2006, pp. 517–530 (cit. on p. 52).

[Bra+17] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother: *DSAC-Differentiable RANSAC for Camera Localization*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6684–6692 (cit. on pp. 7, 17, 84).

[Bra+16] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold, and C. Rother: *Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 3364–3372 (cit. on p. 18).

[BR18] E. Brachmann and C. Rother: *Learning Less is More – 6D Camera Localization via 3D Surface Regression*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4654–4662 (cit. on pp. 17, 64, 70).

[BR19] E. Brachmann and C. Rother: *Expert Sample Consensus Applied to Camera Re-Localization*. *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 7525–7534 (cit. on p. 95).

[Bre01] L. Breiman: *Random Forests*. *Machine learning*. Vol. 45. 1. 2001, pp. 5–32 (cit. on p. 18).

[Cal+10] M. Calonder, V. Lepetit, C. Strecha, and P. Fua: *BRIEF: Binary Robust Independent Elementary Features*. *European Conference on Computer Vision*. 2010, pp. 778–792 (cit. on p. 5).

[CKM08] R. Castle, G. Klein, and D. W. Murray: *Video-Rate Localization in Multiple Maps for Wearable Augmented Reality*. *IEEE International Symposium on Wearable Computers*. 2008, pp. 15–22 (cit. on p. 5).

[Cig+08] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia: *MeshLab: An Open-Source Mesh Processing Tool*. *Eurographics Italian Chapter Conference*. 2008 (cit. on p. 40).

[CD09] G. Conte and P. Doherty: *Vision-Based Unmanned Aerial Vehicle Navigation Using Geo-Referenced Information*. *EURASIP Journal on Advances in Signal Processing*. 2009, p. 10 (cit. on p. 43).

[CD11] G. Conte and P. Doherty: *A Visual Navigation System for UAS Based on Geo-Referenced Imagery*. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. 38. 1/C22. 2011 (cit. on p. 43).

[Csu+04] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray: *Visual Categorization with Bags of Keypoints*. *Workshop on Statistical Learning in Computer Vision*. Vol. 1. 1-22. 2004, pp. 1–2 (cit. on p. 69).

[CGK15] X. Cui, V. Goel, and B. Kingsbury: *Data Augmentation for Deep Neural Network Acoustic Modeling*. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*. Vol. 23. 9. 2015, pp. 1469–1477 (cit. on p. 51).

[CN08] M. Cummins and P. Newman: *FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance*. *The International Journal of Robotics Research*. Vol. 27. 6. 2008, pp. 647–665 (cit. on p. 5).

[Dav+07] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse: *MonoSLAM: Real-time Single Camera SLAM*. IEEE Transactions on Pattern Analysis & Machine Intelligence 6 (2007), pp. 1052–1067 (cit. on p. 5).

[Den+09] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei: *ImageNet: A Large-Scale Hierarchical Image Database*. *IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255 (cit. on p. 44).

[Dus+19] 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*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8092–8101 (cit. on pp. 6, 17).

[ED06] E. Eade and T. Drummond: *Scalable Monocular SLAM*. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 1. IEEE Computer Society. 2006, pp. 469–476 (cit. on p. 5).

[EKC17] J. Engel, V. Koltun, and D. Cremers: *Direct Sparse Odometry*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 40. 3. 2017, pp. 611–625 (cit. on pp. 42, 43).

[ESC14] J. Engel, T. Schöps, and D. Cremers: *LSD-SLAM: Large-Scale Direct Monocular SLAM*. *European Conference on Computer Vision*. 2014, pp. 834–849 (cit. on pp. 42, 43).

[FFP06] L. Fei-Fei, R. Fergus, and P. Perona: *One-shot Learning of Object Categories*. IEEE Transactions on Pattern Analysis and Machine Intelligence 28.4 (2006), pp. 594–611 (cit. on p. 95).

[Fen+19] M. Feng, S. Hu, M. Ang, and G. H. Lee: *2D3D-MatchNet: Learning to Match Keypoints Across 2D Image and 3D Point Cloud*. *arXiv preprint arXiv:1904.09742*. 2019 (cit. on pp. 7, 17).

[FB81] 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*. Vol. 24. 6. 1981, pp. 381–395 (cit. on p. 17).

[FZ98] A. W. Fitzgibbon and A. Zisserman: *Automatic Camera Recovery for Closed or Open Image Sequences*. *European Conference on Computer Vision*. 1998, pp. 311–326 (cit. on p. 20).

[Fli+95] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker: *Query by Image and Video Content: The QBIC System*. Computer 28.9 (1995), pp. 23–32 (cit. on p. 21).

[Fou] O. S. R. Foundation: *gazebo*. URL: `http://gazebosim.org/` (visited on 10/01/2019) (cit. on p. 54).

[Gao+03] 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*. Vol. 25. 8. 2003, pp. 930–943 (cit. on pp. 5, 64).

[GEB15] L. A. Gatys, A. S. Ecker, and M. Bethge: *A Neural Algorithm of Artistic Style*. *arXiv preprint arXiv:1508.06576*. 2015 (cit. on p. 64).

[Gei+18] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel: *ImageNet-trained CNNs are Biased Towards Texture; Increasing Shape Bias Improves Accuracy and Robustness*. *arXiv preprint arXiv:1811.12231*. 2018 (cit. on p. 49).

[Gha+16] M. Gharbi, G. Chaurasia, S. Paris, and F. Durand: *Deep Joint Demosaicking and Denoising*. *ACM Transactions on Graphics*. Vol. 35. 6. 2016, p. 191 (cit. on p. 51).

[GK96] C. Goller and A. Kuchler: *Learning Task-Dependent Distributed Representations by Backpropagation Through Structure*. *Proceedings of International Conference on Neural Networks*. Vol. 1. 1996, pp. 347–352 (cit. on p. 23).

[Goo+14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio: *Generative Adversarial Nets*. *Advances in Neural Information Processing Systems*. 2014, pp. 2672–2680 (cit. on pp. 12, 35, 36, 64).

[Gor+16] A. Gordo, J. Almazán, J. Revaud, and D. Larlus: *Deep Image Retrieval: Learning Global Representations for Image Search*. *European Conference on Computer Vision*. 2016, pp. 241–257 (cit. on p. 7).

[Goy+17] M. Goyal, P. Rajpura, H. Bojinov, and R. Hegde: *Dataset Augmentation with Synthetic Images Improves Semantic Segmentation*. *National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics*. 2017, pp. 348–359 (cit. on pp. 51, 52, 63).

[GS09] A. Graves and J. Schmidhuber: *Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks*. *Advances in Neural Information Processing Systems*. 2009, pp. 545–552 (cit. on p. 23).

[Gru41] J. A. Grunert: *Das Pothenotische Problem in erweiterter Gestalt nebst über seine Anwendung in der Geodäsie*. *Archiv für Mathematik und Physik*. Vol. 1. 1841, pp. 238–248 (cit. on p. 18).

[GVZ16] A. Gupta, A. Vedaldi, and A. Zisserman: *Synthetic Data for Text Localisation in Natural Images*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 2315–2324 (cit. on pp. 51, 63).

[Gup+15] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik: *Inferring 3D Object Pose in RGB-D Images*. *arXiv preprint arXiv:1502.04652*. 2015 (cit. on p. 52).

[Han+15] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg: *MatchNet: Unifying Feature and Metric Learning for Patch-Based Matching*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3279–3286 (cit. on pp. 5, 17).

[Har+94] B. M. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle: *Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem*. *International Journal of Computer Vision*. Vol. 13. 3. 1994, pp. 331–356 (cit. on pp. 5, 18).

[HS88] C. Harris and M. Stephens: *A combined Corner and Edge Detector*. *Alvey Vision Conference*. Vol. 15. 50. 1988, pp. 10–5244 (cit. on p. 15).

[HZ03] R. Hartley and A. Zisserman: *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003 (cit. on pp. 18, 69).

[He+15] K. He, X. Zhang, S. Ren, and J. Sun: *Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 37. 9. 2015, pp. 1904–1916 (cit. on p. 13).

[Hei+15] J. Heinly, J. L. Schönberger, E. Dunn, and J.-M. Frahm: *Reconstructing the World in Six Days*. *Computer Vision and Pattern Recognition*. 2015 (cit. on p. 8).

[HS97] S. Hochreiter and J. Schmidhuber: *Long Short-Term Memory*. *Neural Computation*. Vol. 9. 8. 1997, pp. 1735–1780 (cit. on pp. 6, 23).

[How13] A. G. Howard: *Some Improvements on Deep Convolutional Neural Network Based Image Classification*. *arXiv preprint arXiv:1312.5402*. 2013 (cit. on p. 13).

[HSY12] Y. Huachao, Z. Shubi, and W. Yongbo: *Robust and Precise Registration of Oblique Images Based on Scale-Invariant Feature Transformation Algorithm*. *IEEE Geoscience and Remote Sensing Letters*. Vol. 9. 4. 2012, pp. 783–787 (cit. on p. 43).

[Ian+16] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer: *SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters and <0.5MB Model Size*. *arXiv:1602.07360 [cs]*. 2016 (cit. on pp. 43–45).

[ISI16] S. Iizuka, E. Simo-Serra, and H. Ishikawa: *Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification*. *ACM Transactions on Graphics*. Vol. 35. 4. 2016, 110:1–110:11 (cit. on p. 63).

[IS15] S. Ioffe and C. Szegedy: *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. *Proceedings of The 32nd International Conference on Machine Learning*. 2015, pp. 448–456 (cit. on p. 45).

[Irs+09] 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, pp. 2599–2606 (cit. on pp. 7, 19).

[Iso+17] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros: *Image-to-Image Translation with Conditional Adversarial Networks*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1125–1134 (cit. on p. 63).

[Iva14] C. Ivarsson: *Combining Street View and Aerial Images to Create Photo-Realistic 3D City Models*. (2014) (cit. on p. 52).

[Jac+91] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton: *Adaptive Mixtures of Local Experts*. Vol. 3. 1. 1991, pp. 79–87 (cit. on p. 95).

[Jad+15] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu: *Spatial Transformer Networks*. *Advances in Neural Information Processing Systems*. 2015, pp. 2017–2025 (cit. on p. 17).

[Jég+10] H. Jégou, M. Douze, C. Schmid, and P. Pérez: *Aggregating Local Descriptors into a Compact Image Representation*. *IEEE Conference on Computer Vision and Pattern Recognition*. 2010, pp. 3304–3311 (cit. on p. 52).

[JAF16] J. Johnson, A. Alahi, and L. Fei-Fei: *Perceptual Losses for Real-Time Style Transfer and Super-Resolution*. *European Conference on Computer Vision*. 2016, pp. 694–711 (cit. on p. 64).

[Jun+19] A. B. Jung, K. Wada, J. Crall, S. Tanaka, J. Graving, S. Yadav, J. Banerjee, G. Vecsei, A. Kraft, J. Borovec, C. Vallentin, S. Zhydenko, K. Pfeiffer, B. Cook, I. Fernández, W. Chi-Hung, A. Ayala-Acevedo, R. Meudec, and M. Laporte: *imgaug*. (2019). URL: `https://github.com/aleju/imgaug` (visited on 04/09/2019) (cit. on p. 13).

[Kar+14] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei: *Large-Scale Video Classification with Convolutional Neural Networks*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 1725–1732 (cit. on p. 51).

[KC16] A. Kendall and R. Cipolla: *Modelling Uncertainty in Deep Learning for Camera Relocalization*. *IEEE International Conference on Robotics and Automation*. 2016, pp. 4762–4769 (cit. on pp. 6, 44).

[KC17] A. Kendall and R. Cipolla: *Geometric Loss Functions for Camera Pose Regression with Deep Learning*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 5974–5983 (cit. on pp. 22, 23, 64).

[KGC15] A. Kendall, M. Grimes, and R. Cipolla: *Posenet: A Convolutional Network for Real-Time 6-DoF Camera Relocalization*. *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2938–2946 (cit. on pp. 5, 21, 28, 37, 39, 44–46, 52, 64, 65).

[KW13] D. P. Kingma and M. Welling: *Auto-Encoding Variational Bayes*. *arXiv preprint arXiv:1312.6114*. 2013 (cit. on p. 36).

[KD91] J. J. Koenderink and A. J. van Doorn: *Affine Structure from Motion*. *Journal of the Optical Society of America A*. Vol. 8. 2. 1991, pp. 377–385 (cit. on p. 5).

[KH04] N. Koenig and A. Howard: *Design and Use Paradigms for Gazebo, an Open-Source Multi-Robot Simulator*. *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 3. 2004, pp. 2149–2154 (cit. on p. 54).

[KSH12] A. Krizhevsky, I. Sutskever, and G. E. Hinton: *ImageNet Classification with Deep Convolutional Neural Networks*. *Advances in Neural Information Processing Systems*. 2012, pp. 1097–1105 (cit. on pp. 13, 43).

[KBP10] Z. Kukelova, M. Bujnak, and T. Pajdla: *Closed-Form Solutions to Minimal Absolute Pose Problems with Known Vertical Direction*. *Asian Conference on Computer Vision*. 2010, pp. 216–229 (cit. on p. 19).

[Lad+17] L. Ladický, S. Jeong, N. Bartolović, M. Pollefeys, and M. Gross: *Physics Forests: Real-Time Fluid Simulation Using Machine Learning*. *SIGGRAPH Real Time Live!* 2017 (cit. on p. 95).

[LeC+98] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner: *Gradient-Based Learning Applied to Document Recognition*. *Proceedings of the IEEE*. Vol. 86. 11. 1998, pp. 2278–2324 (cit. on p. 33).

[LBC17] J. Lemley, S. Bazrafkan, and P. Corcoran: *Smart Augmentation Learning an Optimal Data Augmentation Strategy*. *IEEE Access*. Vol. 5. 2017, pp. 5858–5869 (cit. on pp. 51, 63).

[LMF08] V. Lepetit, F. Moreno-Noguer, and P. Fua: *EPnP: An Accurate O(n) Solution to the PnP Problem*. *International Journal of Computer Vision*. Vol. 81. 2. 2008, p. 155 (cit. on pp. 5, 19).

[LCS11] S. Leutenegger, M. Chli, and R. Siegwart: *BRISK: Binary Robust Invariant Scalable Keypoints*. *IEEE International Conference on Computer Vision*. 2011, pp. 2548–2555 (cit. on p. 15).

[LZ13] J. Li and Y. Zhang: *Learning SURF Cascade for Fast and Accurate Object Detection*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 3468–3475 (cit. on p. 52).

[Li+09] Q. Li, G. Wang, J. Liu, and S. Chen: *Robust Scale-Invariant Feature Matching for Remote Sensing Image Registration*. *Geoscience and Remote Sensing Letters*. Vol. 6. 2. 2009, pp. 287–291 (cit. on p. 43).

[LXX12] S. Li, C. Xu, and M. Xie: *A Robust O (n) Solution to the Perspective-n-Point Problem*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 34. 7. 2012, pp. 1444–1450 (cit. on p. 19).

[LSH10] Y. Li, N. Snavely, and D. P. Huttenlocher: *Location Recognition using Prioritized Feature Matching*. *European Conference on Computer Vision*. 2010, pp. 791–804 (cit. on p. 19).

[LA17] B. Limonchik and G. Amdur: *3D Model-Based Data Augmentation for Hand Gesture Recognition*. 2017 (cit. on pp. 13, 52).

[Lin+15] T.-Y. Lin, Yin Cui, S. Belongie, and J. Hays: *Learning Deep Representations for Ground-to-Aerial Geolocalization*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 5007–5015 (cit. on p. 43).

[Llo82] S. Lloyd: *Least Squares Quantization in PCM*. *IEEE Transactions on Information Theory*. Vol. 28. 2. 1982, pp. 129–137 (cit. on p. 69).

[Lom+18] S. Lombardi, J. Saragih, T. Simon, and Y. Sheikh: *Deep Appearance Models for Face Rendering*. Vol. 37. 4. 2018, 68:1–68:13 (cit. on p. 13).

[Low04] D. G. Lowe: *Distinctive Image Features from Scale-Invariant Keypoints*. *International Journal of Computer Vision*. Vol. 60. 2. 2004, pp. 91–110 (cit. on p. 5).

[Luc+16] P. Luc, C. Couprie, S. Chintala, and J. Verbeek: *Semantic Segmentation using Adversarial Networks*. *arXiv preprint arXiv:1611.08408*. 2016 (cit. on p. 35).

[Lyn+15] S. Lynen, T. Sattler, M. Bosse, J. A. Hesch, M. Pollefeys, and R. Siegwart: *Get Out of My Lab: Large-scale, Real-Time Visual-Inertial Localization*. *Robotics: Science and Systems*. 2015 (cit. on p. 5).

[MHN13] A. L. Maas, A. Y. Hannun, and A. Y. Ng: *Rectifier Nonlinearities Improve Neural Network Acoustic Models*. *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*. 2013 (cit. on p. 45).

[Mai+10] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger: *Adaptive and Generic Corner Detection Based on the Accelerated Segment Test*. *European Conference on Computer vision*. 2010, pp. 183–196 (cit. on p. 15).

[Mas+19] I. Masi, A. T. Trân, T. Hassner, G. Sahin, and G. Medioni: *Face-Specific Data Augmentation for Unconstrained Face Recognition*. *International Journal of Computer Vision*. Vol. 127. 6-7. 2019, pp. 642–667 (cit. on p. 13).

[MS15] D. Maturana and S. Scherer: *Voxnet: A 3D Convolutional Neural Network for Real-Time Object Recognition*. *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2015, pp. 922–928 (cit. on pp. 51, 63).

[MP43] W. S. McCulloch and W. Pitts: *A Logical Calculus of the Ideas Immanent in Nervous Activity*. *The Bulletin of Mathematical Biophysics*. Vol. 5. 4. 1943, pp. 115–133 (cit. on p. 25).

[MMB04] C. McGlove, E. Mikhail, and J. Bethel: *Manual of Photogrammetry*. *American Society for Photogrammetry and Remote Sensing*. 2004 (cit. on p. 18).

[MSN05] J. Michels, A. Saxena, and A. Y. Ng: *High Speed Obstacle Avoidance Using Monocular Vision and Reinforcement Learning*. *Proceedings of the 22nd International Conference on Machine learning*. 2005, pp. 593–600 (cit. on p. 52).

[MQV95] R. Mohr, L. Quan, and F. Veillon: *Relative 3D Reconstruction Using Multiple Uncalibrated Images*. The International Journal of Robotics Research 14.6 (1995), pp. 619–632 (cit. on p. 20).

[Mol+15] P. Molchanov, S. Gupta, K. Kim, and J. Kautz: *Hand Gesture Recognition with 3D Convolutional Neural Networks*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2015, pp. 1–7 (cit. on pp. 51, 52, 63).

[Moo+16] K. Moo Yi, Y. Verdie, P. Fua, and V. Lepetit: *Learning to Assign Orientations to Feature Points*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 107–116 (cit. on pp. 16, 17).

[ML09] M. Muja and D. G. Lowe: *Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration*. *International Conference on Computer Vision Theory and Application*. INSTICC Press, 2009, pp. 331–340 (cit. on p. 69).

[MJ18] M. S. Müller and B. Jutzi: *UAS Navigation with SqueezePoseNet – Accuracy Boosting for Pose Regression by Data Augmentation*. *Multidisciplinary Digital Publishing Institute – Drones*. Vol. 2. 1. 2018, pp. 7–27 (cit. on pp. 9, 12, 64).

[MMJ18] M. S. Müller, A. Metzger, and B. Jutzi: *CNN-Based Initial Localization Improved by Data Augmentation*. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. IV-1. 2018, pp. 117–224 (cit. on pp. 9, 12, 63).

[Mül+19] M. S. Müller, T. Sattler, M. Pollefeys, and B. Jutzi: *Image-to-Image Translation for Enhanced Feature Matching, Image Retrieval and Visual Localization*. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. IV-2/W7. 2019, pp. 111–119 (cit. on pp. 10, 12, 14).

[MUJ17] M. S. Müller, S. Urban, and B. Jutzi: *SqueezePoseNet: Image Based Pose Regression with Small Convolutional Neural Networks for Real Time UAS Navigation*. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. IV-2/W3. 2017, pp. 49–57 (cit. on pp. 5, 6, 9, 64).

[MV16] M. S. Müller and T. Voegtle: *Determination of Steering Wheel Angles during Car Alignment by Image Analysis Methods*. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. XLI-B5. 2016, pp. 77–83 (cit. on p. 43).

[MMT15] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos: *ORB-SLAM: A Versatile and Accurate Monocular SLAM System*. *IEEE Transactions on Robotics*. Vol. 31. 5. 2015, pp. 1147–1163 (cit. on pp. 42, 43, 52).

[Nad+19] U. Nadeem, M. A. Jalwana, M. Bennamoun, R. Togneri, and F. Sohel: *Direct Image to Point Cloud Descriptors Matching for 6-DOF Camera Localization in Dense 3D Point Cloud*. *arXiv preprint arXiv:1906.06064*. 2019 (cit. on p. 18).

[NB17] T. Naseer and W. Burgard: *Deep Regression for Monocular Camera-based 6-DoF Global Localization in Outdoor Environments*. *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2017, pp. 1525–1530 (cit. on p. 51).

[Nas51] J. Nash: *Non-Cooperative Games*. *Annals of Mathematics*. 1951, pp. 286–295 (cit. on p. 36).

[Ng+15] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici: *Beyond Short Snippets: Deep Networks for Video Classification*. *IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 4694–4702 (cit. on pp. 51, 63).

[NS06] D. Nister and H. Stewenius: *Scalable Recognition with a Vocabulary Tree*. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 2. 2006, pp. 2161–2168 (cit. on p. 7).

[Nvi17] Nvidia: *Nvidia® Tegra ™Multi-Processor Architecture*. 2017. URL: `http://www.nvidia.com/docs/io/90715/tegra_multiprocessor_architecture_white_paper_final_v1.1.pdf` (visited on 10/02/2019) (cit. on p. 42).

[Pal+19] D. Palossi, A. Loquercio, F. Conti, E. Flamand, D. Scaramuzza, and L. Benini: *A 64mW DNN-Based Visual Navigation Engine for Autonomous Nano-Drones*. *IEEE Internet of Things Journal*. 2019 (cit. on p. 42).

[PVZ15] O. M. Parkhi, A. Vedaldi, and A. Zisserman: *Deep Face Recognition*. *Proceedings of the British Machine Vision Conference*. Vol. 1. 3. 2015, pp. 41.1–41.12 (cit. on pp. 51, 63).

[Pen+15] X. Peng, B. Sun, K. Ali, and K. Saenko: *Learning Deep Object Detectors from 3D Models*. *IEEE International Conference on Computer Vision*. 2015, pp. 1278–1286 (cit. on pp. 13, 51, 52, 63).

[PW17] L. Perez and J. Wang: *The Effectiveness of Data Augmentation in Image Classification Using Deep Learning*. *arXiv preprint arXiv:1712.04621*. 2017 (cit. on p. 12).

[Phi+07] 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, pp. 1–8 (cit. on p. 7).

[Pho] PhotoModeler Technologies: *PhotoModeler*. URL: `www.photomodeler.com` (visited on 10/14/2019) (cit. on p. 54).

[Pol+00] M. Pollefeys, R. Koch, M. Vergauwen, and L. Van Gool: *Automated Reconstruction of 3D Scenes from Sequences of Images*. *ISPRS Journal of Photogrammetry and Remote Sensing*. Vol. 55. 4. 2000, pp. 251–267 (cit. on p. 52).

[PMN18] H. Porav, W. Maddern, and P. Newman: *Adversarial Training for Adverse Conditions: Robust Metric Localisation Using Appearance Transfer*. *IEEE International Conference on Robotics and Automation*. 2018, pp. 1011–1018 (cit. on p. 64).

[PY08] C. Poullis and S. You: *Photorealistic Large-Scale Urban City Model Reconstruction*. *IEEE Transactions on Visualization and Computer Graphics*. Vol. 15. 4. 2008, pp. 654–669 (cit. on p. 52).

[QL99] L. Quan and Z. Lan: *Linear n-Point Camera Pose Determination*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 21. 8. 1999, pp. 774–780 (cit. on p. 5).

[Ree+16] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee: *Generative Adversarial Text to Image Synthesis*. *arXiv preprint arXiv:1605.05396*. 2016 (cit. on p. 35).

[RD06a] G. Reitmayr and T. Drummond: *Going Out: Robust Model-Based Tracking for Outdoor Augmented Reality*. *Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality*. 2006, pp. 109–118 (cit. on p. 43).

[RS16] G. Rogez and C. Schmid: *MoCap-Guided Data Augmentation for 3D Pose Estimation in the Wild*. *Advances in Neural Information Processing Systems 29*. 2016, pp. 3108–3116 (cit. on pp. 51, 52, 63).

[Ros] A. Rosebrock: *Image Search Enginge*. URL: `https://www.pyimagesearch.com/2014/12/01/complete-guide-building-image-search-engine-python-opencv/` (cit. on p. 57).

[Ros58] F. Rosenblatt: *The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain*. *Psychological Review*. Vol. 65. 6. 1958, pp. 386–408 (cit. on p. 25).

[RD06b] E. Rosten and T. Drummond: *Machine Learning for High-Speed Corner Detection*. *European Conference on Computer Vision*. 2006, pp. 430–443 (cit. on p. 5).

[Rub+11] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski: *ORB: An Efficient Alternative to SIFT or SURF*. *IEEE International Conference on Computer Vision*. Vol. 1. 2011, pp. 2564–2571 (cit. on p. 5).

[RHM97] Y. Rui, T. S. Huang, and S. Mehrotra: *Content-Based Image Retrieval with Relevance Feedback in MARS*. *Proceedings of International Conference on Image Processing*. Vol. 2. 1997, pp. 815–818 (cit. on p. 21).

[RHW86] D. E. Rumelhart, G. E. Hinton, and R. J. Williams: *Learning Representations by Back-Propagating Errors*. *Nature*. Vol. 323. 6088. 1986, pp. 533–536 (cit. on p. 28).

[Rus+15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei: *ImageNet Large Scale Visual Recognition Challenge*. Vol. 115. 3. 2015, pp. 211–252 (cit. on p. 44).

[SLK11] T. Sattler, B. Leibe, and L. Kobbelt: *Fast Image-Based Localization Using Direct 2D-to-3D Matching*. *International Conference on Computer Vision*. 2011, pp. 667–674 (cit. on p. 5).

[SLK12] T. Sattler, B. Leibe, and L. Kobbelt: *Improving Image-Based Localization by Active Correspondence Search*. *European Conference on Computer Vision*. 2012, pp. 752–765 (cit. on p. 5).

[SLK16] 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*. Vol. 39. 9. 2016, pp. 1744–1756 (cit. on pp. 5, 52).

[Sat+18a] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, F. Kahl, and P. Tomas: *Benchmarking 6DOF Urban Visual Localization in Changing Conditions*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8601–8610 (cit. on pp. 17, 52, 64).

[Sat+18b] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixé: *Understanding the Limitations of CNN-Based Absolute Camera Pose Regression*. *IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3302–3312 (cit. on p. 89).

[SM97] C. Schmid and R. Mohr: *Local Grayvalue Invariants for Image Retrieval*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 19. 5. 1997, pp. 530–535 (cit. on pp. 5, 52).

[Sch+17] J. L. Schönberger, H. Hardmeier, T. Sattler, and M. Pollefeys: *Comparative Evaluation of Hand-Crafted and Learned Local Features*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1482–1491 (cit. on p. 6).

[Sch+16] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys: *Pixelwise View Selection for Unstructured Multi-View Stereo*. *European Conference on Computer Vision*. 2016, pp. 501–518 (cit. on pp. 20, 65).

[SF16] J. L. Schönberger and J.-M. Frahm: *Structure-from-Motion Revisited*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 4104–4113 (cit. on pp. 5, 20, 52, 65).

[SAS07] P. Scovanner, S. Ali, and M. Shah: *A 3-Dimensional SIFT Descriptor and Its Application to Action Recognition*. *Proceedings of the 15th ACM International Conference on Multimedia*. 2007, pp. 357–360 (cit. on p. 18).

[SJ06] S. Se and P. Jasiobedzki: *Photo-Realistic 3D Model Reconstruction*. *Proceedings of the IEEE International Conference on Robotics and Automation*. 2006, pp. 3076–3082 (cit. on p. 52).

[Sha+14] Q. Shan, C. Wu, B. Curless, Y. Furukawa, C. Hernandez, and S. M. Seitz: *Accurate Geo-Registration by Ground-to-Aerial Image Matching*. *2nd International Conference on 3D Vision*. Vol. 1. 2014, pp. 525–532 (cit. on p. 52).

[Sha+15] A. Sharif Razavian, J. Sullivan, A. Maki, and S. Carlsson: *A Baseline for Visual Instance Retrieval with Deep Convolutional Networks*. *International Conference on Learning Representations*. 2015 (cit. on p. 52).

[SN18] S. Sharma and V. P. Namboodiri: *No Modes Left Behind: Capturing the Data Distribution Effectively using GANs*. *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018, pp. 4042–4049 (cit. on p. 51).

[ST94] J. Shi and C. Tomasi: *Good Features to Track*. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 1994, pp. 593–600 (cit. on p. 15).

[Sho+13a] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon: *Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 2930–2937 (cit. on p. 18).

[Sho+13b] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon: *Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images*. *The IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 2930–2937 (cit. on p. 37).

[Shr+17a] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb: *Learning From Simulated and Unsupervised Images Through Adversarial Training*. *IEEE Conference on Computer Vision and Pattern Recognition*. 2017 (cit. on p. 64).

[Shr+17b] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb: *Learning from Simulated and Unsupervised Images through Adversarial Training*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 2107–2116 (cit. on p. 14).

[Sib+13] D. Sibbing, T. Sattler, B. Leibe, and L. Kobbelt: *SIFT-Realistic Rendering*. *2013 International Conference on 3D Vision*. 2013, pp. 56–63 (cit. on p. 52).

[SSP03] P. Y. Simard, D. Steinkraus, and J. C. Platt: *Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis*. *International Conference on Document Analysis and Recognition*. Vol. 3. 2003 (cit. on p. 13).

[SVZ14] K. Simonyan, A. Vedaldi, and A. Zisserman: *Learning Local Feature Descriptors Using Convex Optimisation*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 36. 8. 2014, pp. 1573–1585 (cit. on pp. 16, 17).

[SZ14a] K. Simonyan and A. Zisserman: *Two-Stream Convolutional Networks for Action Recognition in Videos*. *Advances in Neural Information Processing Systems*. 2014, pp. 568–576 (cit. on p. 13).

[SZ14b] K. Simonyan and A. Zisserman: *Very Deep Convolutional Networks for Large-Scale Image Recognition*. *arXiv:1409.1556*. 2014 (cit. on pp. 43–45).

[Sim+15] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer: *Discriminative Learning of Deep Convolutional Feature Point Descriptors*. *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 118–126 (cit. on pp. 16, 17).

[SJM13] S. P. Singh, K. Jain, and V. R. Mandla: *Virtual 3D City Modeling: Techniques and Applications*. *ISPRS International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. 2. 2013, pp. 73–91 (cit. on p. 52).

[SB11] I. Sipiran and B. Bustos: *Harris 3D: A Robust Extension of the Harris Operator for Interest Point Detection on 3D Meshes*. *The Visual Computer*. Vol. 27. 11. 2011, pp. 963–976 (cit. on p. 18).

[SGS10] M. Stark, M. Goesele, and B. Schiele: *Back to the Future: Learning Shape Models from 3D CAD Data*. *British Machine Vision Conference*. Vol. 2. 4. 2010, pp. 5–15 (cit. on p. 52).

[Su+15] H. Su, C. R. Qi, Y. Li, and L. J. Guibas: *Render for CNN: Viewpoint Estimation in Images Using CNNs Trained with Rendered 3D Model Views*. *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 2686–2694 (cit. on pp. 13, 52).

[SVL14] I. Sutskever, O. Vinyals, and Q. V. Le: *Sequence to Sequence Learning with Neural Networks*. *Advances in Neural Information Processing Systems*. 2014, pp. 3104–3112 (cit. on p. 23).

[SW96] D. L. Swets and J. J. Weng: *Using Discriminant Eigenfeatures for Image Retrieval*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 18. 8. 1996, pp. 831–836 (cit. on p. 52).

[Sze+15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich: *Going Deeper with Convolutions*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 1–9 (cit. on p. 44).

[TN17] L. Taylor and G. Nitschke: *Improving Deep Learning Using Generic Data Augmentation*. *arXiv preprint arXiv:1708.06020*. 2017 (cit. on p. 12).

[Tor+15] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla: *24/7 Place Recognition by View Synthesis*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 1808–1817 (cit. on p. 53).

[TZ00] P. H. Torr and A. Zisserman: *MLESAC: A New Robust Estimator with Application to Estimating Image Geometry*. *Computer Vision and Image Understanding*. Vol. 78. 1. 2000, pp. 138–156 (cit. on pp. 18, 69).

[Tri+99] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon: *Bundle Adjustment – A Modern Synthesis*. *International Workshop on Vision Algorithms*. 1999, pp. 298–372 (cit. on p. 20).

[Tu05] Z. Tu: *Probabilistic Boosting-Tree: Learning Discriminative Models for Classification, Recognition, and Clustering*. *Tenth IEEE International Conference on Computer Vision*. Vol. 2. 1589-1596. 2005, p. 2 (cit. on p. 51).

[Ull79] S. Ullman: *The interpretation of Structure from Motion*. *Proceedings of the Royal Society of London. Series B. Biological Sciences*. Vol. 203. 1153. 1979, pp. 405–426 (cit. on p. 19).

[UWS09] M. Ulrich, C. Wiedemann, and C. Steger: *CAD-Based Recognition of 3D Objects in Monocular Images*. *International Conference on Robotics and Automation*. Vol. 9. 2009, pp. 1191–1198 (cit. on p. 43).

[URH16] J. Unger, F. Rottensteiner, and C. Heipke: *Integration of a Generalised Building Model Into the Pose Estimation of UAS Images*. *23rd International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences Congress*. 2016 (cit. on p. 43).

[Uni] C. University: *BigSFM: Reconstructing the World from Internet Photos*. URL: `http://www.cs.cornell.edu/projects/bigsfm/` (visited on 10/04/2019) (cit. on p. 8).

[UJ17] S. Urban and B. Jutzi: *LaFiDa - A Laserscanner Multi-Fisheye Camera Dataset*. *Journal of Imaging*. Vol. 3. 1. 2017, p. 5 (cit. on p. 37).

[Urb+13] S. Urban, J. Leitloff, S. Wursthorn, and S. Hinz: *Self-Localization of a Multi-Fisheye Camera Based Augmented Reality System in Textureless 3D Building Models*. Vol. 2. 2013, pp. 43–48 (cit. on p. 43).

[Ver+15] Y. Verdie, K. Yi, P. Fua, and V. Lepetit: *TILDE: A Temporally Invariant Learned Detector*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 5279–5288 (cit. on pp. 16, 17).

[Wal+17] F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck, and D. Cremers: *Image-Based Localization Using LSTMs for Structured Feature Correlation*. *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 627–637 (cit. on pp. 6, 21, 23, 44, 64).

[Wer74] P. Werbos: *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. *Ph.D. Dissertation, Committee on Applied Mathematics, Harvard University*. 1974 (cit. on p. 28).

[Wer82] P. Werbos: *Applications of Advances in Nonlinear Sensitivity Analysis*. *Systems Modeling and Optimization*. 1982, pp. 1550–1560 (cit. on p. 28).

[Wu13] C. Wu: *Towards Linear Time Incremental Structure from Motion*. *2013 International Conference on 3D Vision*. 2013, pp. 127–134 (cit. on pp. 5, 20).

[Yan+18] B. Yang, S. Rosa, A. Markham, N. Trigoni, and H. Wen: *3D Object Dense Reconstruction from a Single Depth View*. *arXiv preprint arXiv:1802.00411*. Vol. 1. 2. 2018, p. 6 (cit. on p. 52).

[Yi+16] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua: *LIFT: Learned Invariant Feature Transform*. *European Conference on Computer Vision*. 2016, pp. 467–483 (cit. on pp. 6, 17).

[YH09] J.-C. Yoo and T. H. Han: *Fast Normalized Cross-Correlation*. Circuits, Systems and Signal Processing 28.6 (2009), pp. 819–843 (cit. on p. 5).

[Yua+19] L. Yuan, C. Ruan, H. Hu, and D. Chen: *Image Inpainting Based on Patch-GANs*. *IEEE Access*. Vol. 7. 2019, pp. 46411–46421 (cit. on p. 35).

[Zho+16] B. Zhou, A. Khosla, A. Lapedriza, A. Torralba, and A. Oliva: *Places: An image Database for Deep Scene Understanding*. *arXiv:1610.02055*. 2016 (cit. on p. 46).

[Zho+14] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva: *Learning Deep Features for Scene Recognition Using Places Database*. *Advances in Neural Information Processing Systems*. 2014, pp. 487–495 (cit. on p. 46).

[ZYS09] H. Zhou, Y. Yuan, and C. Shi: *Object Tracking using SIFT Features and Mean Shift*. *Computer Vision and Image Understanding*. Vol. 113. 3. 2009, pp. 345–352 (cit. on p. 52).

[Zhu+17] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros: *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*. *IEEE International Conference on Computer Vision*. 2017 (cit. on pp. 14, 35, 64, 66).

List of publications

[Hil+19] M. Hillemann, M. Weinmann, M. S. Müller, and B. Jutzi: *Automatic Extrinsic Self-Calibration of Mobile Mapping Systems Based on Geometric 3D Features*. Vol. 11. 16. 2019, pp. 1955–1980.

[MJ18] M. S. Müller and B. Jutzi: *UAS Navigation with SqueezePoseNet – Accuracy Boosting for Pose Regression by Data Augmentation*. *Multidisciplinary Digital Publishing Institute – Drones*. Vol. 2. 1. 2018, pp. 7–27.

[MMJ18] M. S. Müller, A. Metzger, and B. Jutzi: *CNN-Based Initial Localization Improved by Data Augmentation*. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. IV-1. 2018, pp. 117–224.

[Mül+19] M. S. Müller, T. Sattler, M. Pollefeys, and B. Jutzi: *Image-to-Image Translation for Enhanced Feature Matching, Image Retrieval and Visual Localization*. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. IV-2/W7. 2019, pp. 111–119.

[MUJ17] M. S. Müller, S. Urban, and B. Jutzi: *SqueezePoseNet: Image Based Pose Regression with Small Convolutional Neural Networks for Real Time UAS Navigation*. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. IV-2/W3. 2017, pp. 49–57.

[MV16] M. S. Müller and T. Voegtle: *Determination of Steering Wheel Angles during Car Alignment by Image Analysis Methods*. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. Vol. XLI-B5. 2016, pp. 77–83.

[Rös+16] N. Rösch, S. Mann, P. Runge, S. Ziegler, and M. S. Müller: *Lebendige Technikgeschichte – Erste Erfahrungen mit dem nachgebauten "Novum Instrumentum Geometricum" von Leonhard Zubler*. *VDVMagazin*. Vol. 1. 2016, pp. 40–46.

[Wei+17c] M. Weinmann, M. S. Müller, M. Hillemann, N. Reydel, S. Hinz, and B. Jutzi: *Point Cloud Analysis for UAV-Borne Laser Scanning with Horizontally and Vertically Oriented Line Scanners – Concept and Frist Results*. Vol. 42. 2017, pp. 399–406.

Appendix

A Camera Pose

The pose of an object (or camera) specifies its position (or camera center) and orientation in space. The position and orientation of an object in space can be directly described by the translation and rotation of this object referenced to a coordinate frame in the same space. In a three-dimensional space the position or center $\mathbf{c} \in \mathbb{R}^3$ is defined by three parameters denoted as c_x-, c_y-, and c_z-coordinates, which correspond to three DoF concerning the freedom of movement for a specific object in space. Therefore, parameters c_x-, c_y-, and c_z describe the translation of an object on the corresponding X-, Y- and Z-axis of a coordinate frame.

The rotation $\mathbf{R} \in SO(3)$ is minimally parameterized three parameters, whereas $SO(3)$ is the 3D rotation group of all rotations about the origin of a $\mathbb{R}^3$ Euclidean space. Such a parametrization can be given by the Euler Angles (Section A.1) parametrized by the parameters ϕ, θ and ψ. However, this minimal parametrization has disadvantages concerning gimbal lock and interpolation. Quaternion parameterization overcomes this issues and is therefore utilized in computer vision algorithms more widely (Section A.2). A parameterization by quaternions consists of four parameters denoted in the following as w, q_1, q_2 and q_3, which correspond to three DoF of the rotation for a specific object. The three parameters of translation and the (at least) three parameters of rotation are collectively defining the six parameters of the pose. The camera pose is therefore defined with six DoF. In this thesis the terms camera pose and image pose are referred synonymously, whereas the image pose is representing the pose of an image acquired from a camera with the identical pose. The higher level coordinate frame is termed world frame.

The camera's position or camera center $\mathbf{c}$ in world coordinates and the camera's orientation $\mathbf{R}_\mathrm{c}$ can be expressed by its translation and rotation as

$$\mathbf{c} = -\mathbf{R}^{-1}\mathbf{t} \ , \ \mathbf{R}_\mathrm{c} = \mathbf{R}^\mathrm{T} \ , \tag{1}$$

whereas $\mathbf{t}$ is the translation vector and $\mathbf{R}$ is the 3×3 rotation matrix

$$\mathbf{t} = [t_x, t_y, t_z, 1]^\mathrm{T} \ , \ \mathbf{R} = \begin{bmatrix} r11 & r12 & r13 \\ r21 & r22 & r23 \\ r31 & r32 & r33 \end{bmatrix} . \tag{2}$$

Moreover, the rotation $\mathbf{R}$ and the translation $\mathbf{t}$ form the extrinsic matrix $\mathbf{E}$

$$\mathbf{E} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\mathrm{T} & 1 \end{bmatrix} = \begin{bmatrix} r11 & r12 & r13 & t_x \\ r21 & r22 & r23 & t_y \\ r31 & r32 & r33 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{3}$$

Given the extrinsic matrix, a transformation from the camera frame to world frame can be expressed. A transformation from the world frame to the camera frame is given by $\mathbf{E}^{-1}$

$$\mathbf{E}^{-1} = \begin{bmatrix} \mathbf{R}^{\mathrm{T}} & -\mathbf{R}^{\mathrm{T}}\mathbf{c} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix} . \tag{4}$$

where $\mathbf{R}^{-1} = \mathbf{R}^T$ from $\mathbf{R}$ being orthogonal. With the extrinsic matrix $\mathbf{E}$ respectively $\mathbf{E}^{-1}$, a point in the world frame $\mathbf{X}_W = [x_W, y_W, z_W, 1]^T$ can be transformed to a point in the camera frame $\mathbf{X}_C$ or vice versa

$$\mathbf{X}_C = \mathbf{E}^{-1}\mathbf{X}_W = \begin{bmatrix} \mathbf{R}^{\mathrm{T}} & -\mathbf{R}^{\mathrm{T}}\mathbf{c} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix} [x_W, y_W, z_W, 1]^{\mathrm{T}} . \tag{5}$$

Given multiple corresponding points in the world and image frame, Equation 5 can be solved for the camera pose E. This is used for camera pose estimation in feature-based methods solving the PnP problem or SfM (Section 2.3. Figure 1 depicts the relation $[\mathbf{R}, \mathbf{t}]$ between the camera center $\mathbf{C}_0$ and the world reference frame $\mathbf{W}_0$ over 3D points in the scene $\mathbf{X}$.

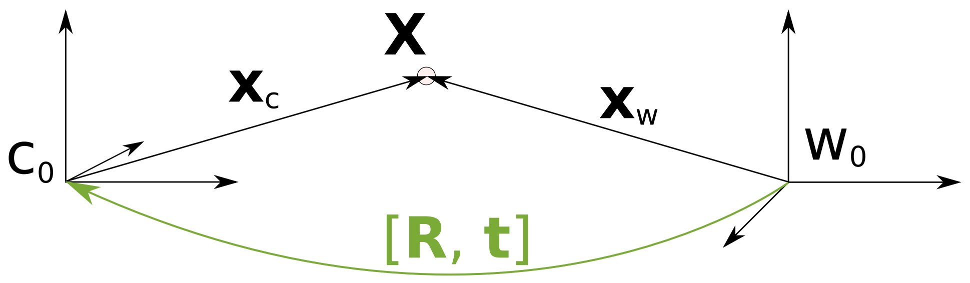


Figure 1: A point $\mathbf{X}$ in the world reference frame $\mathbf{W}_0$ can be described by its world coordinates $\mathbf{X}_W$. The same point can be found in a camera coordinate system $\mathbf{C}_0$ and is described by the vector X_C. The camera pose is defined by the extrinsics matrix $\mathbf{E}$ composed by the camera's rotation $\mathbf{R}$ and translation $\mathbf{t}$. The point $\mathbf{X}_C$ in the camera frame can be transformed from point $\mathbf{X}_W$ by multiplying $\mathbf{X}_W$ by $\mathbf{E}^{-1}$ (Equation 5).

Although, a homogeneous representation is a practical notation for camera poses and transformation, it contains more parameters than required. A minimum of six parameters defines a camera pose or a rigid transformation in $\mathbb{R}^3$ which relates to six DoF. Three parameters are already well encoded by the translation component $\mathbf{t}$. The rotation matrix $\mathbf{R}$ however contains nine elements. Notations by Euler angles or quaternions which manage to describe rotations with three respectively four parameters follow in the subsequent sections.

A.1 Euler Angles

Euler angles are the most intuitive expression of rotations and describe the orientation of an object in $\mathbb{R}^3$ with respect to a reference frame. With three parameters the euler angles represent a minimal parameterization while any orientation can be achieved by composing elemental rotations. There are twelve different orders of rotation, where the so-called x-rotation (z-x-z) is the most common definition. Let ϕ, θ, ψ be the euler angles that represent the rotation about the respective coordinate axis. The rotation matrices that represent such a rotation about are then composed as

$$R_Z(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} , \quad R_X(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} ,$$
$$R_Z(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} . \tag{6}$$

The first rotation is about the z-axis using $R_Z(\phi)$, the second rotation is about the former x-axis by $R_X(\theta)$ and the third rotation is about the former z-axis by $R_Z(\psi)$. Three rotations are always sufficient to reach any state of orientation. However, disadvantages of such a representation is the so-called gimbal lock problem and interpolation issues. Gimbal lock is a degenerate state of rotation and occurs when two axis are rotated into a parallel constellation reducing the DoF to two, thus making a unique determination of the angles impossible. This occurs when the second rotation has a value of $\frac{\pi}{2} + n\pi$ where $n \in \mathbb{Z}$. The interpolation problem occurs due to the numerical range of the angles from $[0, ..., 2\pi]$.

While euler angles have the advantages of an easy human interpretation of rotation and representing a minimum parameterization of rotation. While this is an advantage over quaternions in terms of optimization processes, the mathematical drawbacks predominate. Quaternions avoid the drawbacks and are introduced in the next following.

A.2 Quaternions

A parametrization by quaternions consists of four parameters which correspond to three DoF of the rotation for a specific object. A quaternion has one real part w and three imaginary parts $q_1 \cdot i$, $q_2 \cdot j$, $q_3 \cdot k$. The quaternion $\mathbf{q} \in \mathbb{H}$, where $\mathbf{q} \in \mathbb{H}$ is the set of quaternions, is then denoted as

$$\mathbf{q} = w + q_1 \cdot i + q_2 \cdot j + q_3 \cdot k , \tag{7}$$

where w, q_1, q_2, q_3 are real numbers and i, j, k can be interpreted as the unit vectors pointing along the three spatial axis X, Y, Z. For simplicity only the real part is often denoted and the imaginary multipliers i, j and k are assumed by the order of position leading to

$$\mathbf{q} = w + q_1 + q_2 + q_3 . \tag{8}$$

w represents the scale of the quaternion, whereas q_1, q_2, q_3 denote the vector part. Four parameters to define a rotation in $\mathbb{R}^3$ is an over-parametrization. However, a parametrization by quaternions has computational advantages over an Euler angle parametrization since it is less accurate when the rotation is incrementally estimated or interpolated. Additionally arbitrary 4D values are easier to normalize to unit length compared to the ortho-normalization of rotation matrices.

Furthermore, the difference between two normalized quaternions $\mathbf{q}_{norm}$ and $\hat{\mathbf{q}}_{norm}$ is computed as

$$\Theta = 2 \arccos \left(\mathbf{q}_{norm} \cdot \hat{\mathbf{q}}_{norm} \right) , \tag{9}$$

where $\mathbf{q}^{*}_{norm}$ are normalized quaternions given by a ℓ^2 normalization

$$\mathbf{q}_{norm} = \frac{\mathbf{q}}{\|\mathbf{q}\|_2} , \tag{10}$$

with $\|\mathbf{q}\|_2$ as the ℓ^2-norm

$$\|\mathbf{q}\|_2 = \sqrt{w^2 + q_1^2 + q_2^2 + q_3^2} . \tag{11}$$

Acknowledgements

Danke Mama und Papa. Danke Shannen.

Karlsruhe, April 2020 Markus Müller