Thesis for the Degree of Licentiate of Engineering

Towards Robust Visual Localization in Challenging Conditions

Carl Toft

CHALMERS

Department of Electrical Engineering
Chalmers University of Technology
Göteborg, Sweden 2019

Towards Robust Visual Localization in Challenging Conditions
Carl Toft

Technical report number: R004/2019
ISSN 1403-266X

Department of Electrical Engineering
Computer Vision and Image Analysis Group
Chalmers University of Technology
SE–412 96 Göteborg, Sweden

Typeset by the author using LaTeX.

Chalmers Reproservice
Göteborg, Sweden 2019

Towards Robust Visual Localization in Challenging Conditions
Carl Toft
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Visual localization is a fundamental problem in computer vision, with a multitude of applications in robotics, augmented reality and structure-from-motion. The basic problem is to, based on one or more images, figure out the position and orientation of the camera which captured these images relative to some model of the environment. Current visual localization approaches typically work well when the images to be localized are captured under similar conditions compared to those captured during mapping. However, when the environment exhibits large changes in visual appearance, due to e.g. variations in weather, seasons, day-night or viewpoint, the traditional pipelines break down. The reason is that the local image features used are based on low-level pixel-intensity information, which is not invariant to these transformations: when the environment changes, this will cause a different set of keypoints to be detected, and their descriptors will be different, making the long-term visual localization problem a challenging one.

In this thesis, four papers are included, which present work towards solving the problem of long-term visual localization. Three of the articles present ideas for how semantic information may be included to aid in the localization process: one approach relies only on the semantic information for visual localization, another shows how the semantics can be used to detect outlier feature correspondences, while the third presents a sequential localization algorithm which relies on the consistency of the reprojection of a semantic model, instead of traditional features. The final article is a benchmark paper, where we present three new benchmark datasets aimed at evaluating localization algorithms in the context of long-term visual localization.

Keywords: Visual localization, camera pose estimation, long-term localization, self-driving cars, autonomous vehicles, benchmark

Included publications

Paper I C. Toft, T. Sattler, W. Maddern, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, T. Pajdla and F. Kahl. "Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions". *To be submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI).* Extended version of paper (a).

Paper II C. Toft, E. Stenborg, L. Hammarstrand, L. Brynte, M. Pollefeys, T. Sattler and F. Kahl. "Semantic Match Consistency for Long-Term Visual Localization". *European Conference on Computer Vision (ECCV),* 2018.

Paper III E. Stenborg, C. Toft and L. Hammarstrand. "Long-term Visual Localization Using Semantically Segmented Images". *International Conference on Robotics and Automation (ICRA), 2018*

Paper IV C. Toft, C. Olsson and F. Kahl. "Long-term 3D Localization and Pose from Semantic Labellings". *3D Reconstruction Meets Semantics (3DRMS) Workshop at the International Conference on Computer Vision (ICCV) 2017.*

Subsidiary publications

(a) T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, F. Kahl, and T. Pajdla. "Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions". *Conference on Computer Vision and Pattern Recognition (CVPR) 2018.*

(b) V. Larsson, J. Fredriksson, C. Toft and F. Kahl "Outlier Rejection for Absolute Pose Estimation with Known Orientation". *British Machine Vision Conference (BMVC) 2016* 41–52, 2017.

Contents

II Included Publications

Part I

Introductory Chapters

Chapter 1

Introduction

At the most fundamental level, the field of visual localization aims to answer the question "Where am I?" based on one or more images. This is a problem which humans seem to solve almost effortlessly as we go about our daily tasks: we track our position in the world with little effort, sometimes in previously unseen environments, and successfully use this information to navigate and plan the path to our destination. Getting lost is the exception rather than the norm.

However, as often seems to be the case, tasks which humans find easy and intuitive turn out to be very challenging to find a general algorithmic solution to. The problem of visual localization is no exception.

In order to provide a satisfactory answer this problem, the system needs some internal representation of the world, relative to which the answer can be provided, and it is possible to imagine many different forms in which an answer may be given. For some applications, an answer such as "in the living room" may be sufficient, whereas other applications, such as navigation of autonomous vehicles, may require considerably more precision in the answer. For these applications, the absolute position in terms of x-, y- and $z-$ coordinates, as well as orientation, relative to some coordinate system may be desired.

Providing such a six degree-of-freedom position would require a 3D model of the environment to be constructed beforehand, and the localization would occur with respect to this map. Map construction and representation are thus closely linked to the localization problem. In fact, camera-pose estimation forms a core building block of many 3D reconstruction (often called Structure-from-Motion, or SfM for short) pipelines, where the 3D model is incrementally extended by triangulating the position of cameras, one at a time [1, 2].

Figure 1.1 illustrates the basic goal of the visual localization problem.

While the single-image localization problem is important in its own right,

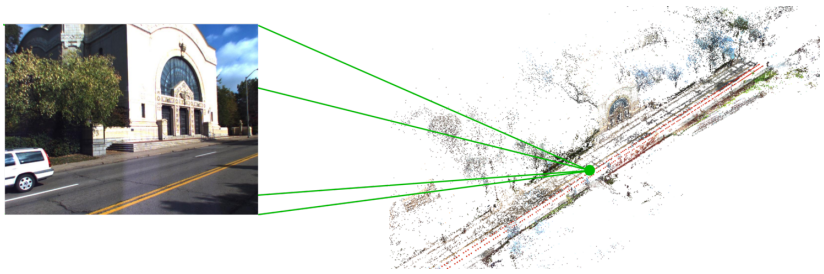


Figure 1.1: The camera pose estimation problem. Given one or more input images, we wish to compute the position of the camera which captured the image, relative to some map representation of the external world. A very common type of map is the three-dimensional point-cloud.

a common scenario in robotics is that of sequential localization where we wish to compute the most likely position of a robot, based on all measurements which have been collected up until the current time. These measurements may contain, for example, a video feed, or a sequence of images from one or more cameras, as well as measurements from inertial measurement units (IMUs) and wheel-speed sensors. The goal is then to combine these measurements to obtain a reliable estimate of the current pose of the vehicle.

The problem of visual localization has received a substantial amount of attention in both the research community and industry the past few years, much due to the surge in interest in autonomous vehicles such as drones and self-driving cars. The camera is often seen as an affordable but information-rich sensor, which might be used instead of, or as a complement to, more expensive sensor setups with Lidars and radars. Automotive grade GPS receivers may be used to aid in positioning, but is often not accurate enough: the positioning error in these kinds of GPS receivers is often on the order of meters [3], much too high for the application. The camera may then be a viable alternative to (or complement to), these GPS receivers. Of course, there exist more accurate (but also considerably more expensive) survey-grade GPS receivers, but these may still be subject to signal acquisition failure in tunnels or indoors. They may also be unreliable in densely populated cities, where there may be no line-of-sight to the GPS satellites in the ”urban canyons”, deteriorating the localization performance as the signal reflects off of the tall buildings on the way down to street-level.

The camera, if its challenges can be overcome, is thus seen as an attractive potential sensor for vehicle localization.

Today, one of the most common pipelines for camera pose estimation utilizes a 3D model in the form of a point cloud, where each point, in ad-

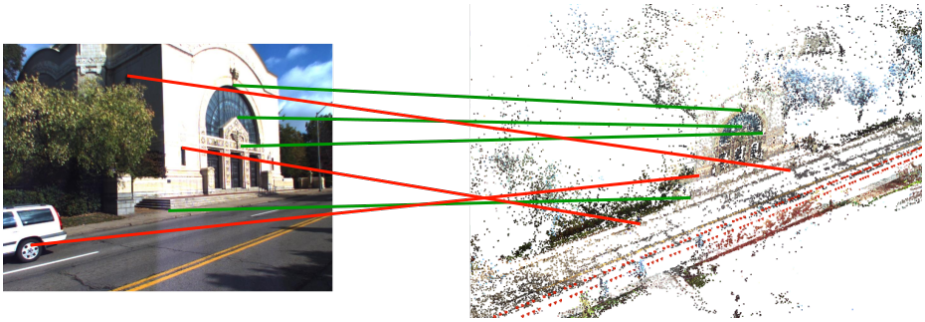


Figure 1.2: Before pose estimation, matches between the query image and the 3D model are established. Some of the matches will typically be incorrect (shown in red). These 2D-3D correspondences are then inputted to a triangulation method which yields the pose of the camera where as many of the 3D matches in the map as possible are projected down onto the corresponding 2D points in the image.

dition to its Cartesian coordinates, also has an associated descriptor vector (often 128 bytes, such as for SIFT-descriptors [4]), which encodes the local appearance of the point as it was seen in the cameras during map building.

In order to localize an image taken somewhere in the map, local image features are extracted from the query image, and 2D-3D correspondences are established between the features in the query image, and the 3D map points. This is typically done by, for each keypoint in the image, finding its approximate nearest neighbour in the map in terms of descriptor distance using a kd-tree search [5]. Using these matches, the pose can then be estimated using a perspective-three-point solver [6, 7], in combination with a robust estimation technique such as RANSAC [8]. Fig. 1.2 illustrates how query features are matched to 3D points in the map.

This purely geometrical approach based on local image appearance typically works well when the query image is taken under similar conditions as the map. However, if the appearance variation is too large, the local image appearance may not be discriminative enough to correctly match the 2D features to the 3D map. This may then yield too few correct matches to accurately estimate the camera pose.

Fig. 1.3 illustrates how the same scene can change in appearance across seasons. If an autonomous vehicle is to employ camera-based navigation over a long period of time, then it should ideally be able to handle these kinds of visual changes. Creation of accurate maps is a time-consuming and expensive process, and creating maps for every single conceivable visual condition is unfeasible. Rather, we desire our localization systems to be robust to these kinds of changes.

Figure 1.3: Two images from the CMU Visual Localization dataset[9], taken from approximately the same position, but during different seasons. While clearly the same spot, matching features based only on local appearance information would yield very few correct matches.

Today, most systems are not able to reliably handle these sorts of visual changes, and developing robust localization methods is a field of active research. It is also the topic of this thesis.

1 Thesis aim and scope

In this thesis we address the problem of long-term visual localization, i.e., localizing images which are taken under conditions very dissimilar to the condition under which the mapping images were captured. Specifically, we address the problem of single-image localization, as well as sequential localization. We try to make the localization pipeline more robust by incorporating higher-level information in the form of semantic segmentations. By also employing a semantically labelled 3D model, we show how this information can be used directly for pose estimation as a replacement for SIFT, and how it can be used as a complement to a SIFT based pipeline by using semantics to correctly identify mismatched correspondences (the red correspondences in Fig. 1.2). We also introduce three new datasets aimed at evaluating long-term visual localization algorithms. We hope these will be of value to the community.

2 Thesis outline

This thesis consists of two parts. In this first part, background material is presented in Chapter 2. This is intended as a ”warm up” containing the necessary background material to comfortably tackle the appended papers at

the end. Readers already familiar with the camera-pose-estimation problem can likely skip this chapter without any loss of comprehension when reading the remainder of the thesis. Chapter 3 summarizes the contents of the four papers, and clarifies the author's contribution to each of them. Lastly, Chapter 4 concludes the first part of the thesis and presents a final outlook on possible future directions for research.

The second part consists of four appended papers, and represents the main content and novel research work of the thesis.

Chapter 2

Background

The main part of this thesis consists of the papers appended in Part II. However, research articles are often quite terse and do not elaborate significantly on the background material since the reader is assumed to already be more or less familiar with it. This can make reading papers in a new area very challenging, since the overall setting in which the paper takes place may not be fully explained.

The purpose of this chapter is to serve as such a warm-up and summarize the relevant background material necessary to understand the contents of the papers included in Part II, and elaborates some more on the general context that may be lacking in the individual papers themselves. The structure of the chapter is as follows: Sec. 1 gives an coarse taxonomy of the visual localization problem, and some common approaches that may be used to solve it. Section 2 introduces what local image features are, how they can be used to establish matches between images, and how they break down in the long-term localization scenario. Section 3 discusses the geometry of camera projection, and how correspondences may be used to calculate camera pose relative to a map. Section 4 explains how image-retrieval based visual localization works. Lastly, Section 5 brings up the topic of semantic segmentations, and suggests how they may potentially aid in visual localization tasks.

1 Visual localization

As mentioned in the introduction, the problem of visual localization is to determine the position of the camera which captured one or more images with respect to a map. There exist many different variations of this problem, with accompanying solutions, depending on what input data is available to aid in the localization (such as the number of images, any prior information

on location, data from additional sensors such as IMU, Lidar, etc) as well as what representation is used for the map.

One may roughly categorize the most common localization methods into two categories [10]: metric and topological. This is not a strict classification, and any given method may fall somewhere in between.

In topological localization, the map is represented as a discrete set of places, often encoded as nodes in a graph. The nodes may then represent places, and adjoining edges correspond to possible paths between these places. The visual localization problem is the to select, from this finite set of nodes, the one which corresponds to the position where the image was captured. Depending on what the graph represents, this may correspond to an answer such as "in the kitchen", or a specific street intersection. This thus corresponds to a discrete classification problem. Finer-grained localization could be obtained by placing nodes densely, and attaching metric coordinates to each node, as done in e.g. [10–12]. Assigning the query image to a node thus also yields approximate metric coordinates for the position of the camera.

This discrete form of visual localization is sometimes referred to as visual place recognition, and is often solved using image retrieval methods, which will be discussed briefly in Sec. 4.

The other class of localization methods are the metric methods. Here, the goal is to output the camera position in metric coordinates, such as latitude, longitude and altitude, or more generally the coordinates with respect to some pre-defined coordinate system. In full six degree-of-freedom camera pose estimation, the goal is to also estimate the three rotational degrees of freedom, in addition to the three translational degrees of freedom, for a total of six real numbers.

Metric localization methods often employ a pre-constructed 3D map of the environment. By associating landmarks detected by the vehicle's sensors to landmarks with known position in the map, it is possible to reason about the position of the vehicle in the map. However, it should be noted that not all metric localization methods need to rely on an explicit 3D map of the environment. Other approaches, such using a neural network to directly regress the six degree-of-freedom pose directly from the image have been explored [13].

For cameras, the landmarks typically consist of point-features detected in the image by an image feature detector. These image features form the backbone of modern 3D reconstruction and visual localization pipelines, and understanding them is central to understanding the shortcomings of current visual localization systems in the long-term localization scenario. We will thus describe these local features more in depth in the following section.

2 Local image features

Image feature (or keypoint) detection and description is one of the most fundamental problems in computer vision, and forms the foundation on which a large body of other methods rest [14]. The purpose of the feature detector is to extract from an image a set of interest points we believe we will be able to redetect in a different image of the same scene, and the purpose of the descriptor is to encode the appearance of the keypoints into a descriptor vector, such that keypoints in the first image can be associated to keypoints in the second image (or map).

If the same set of points can be detected in a different image, we can establish point-correspondences between images, or between an image and a 3D model. Image-to-image correspondences can be used for calculating relative camera poses, which enable subsequent 3D reconstruction of the scene [15]. Image-to-model correspondences allow us to compute the camera pose relative to the model.

In this section we will first briefly discuss keypoint detectors and descriptors, and then have a look at them in the context of long-term visual localization.

2.1 Feature detectors

Feature detectors have been studied since the early days of computer vision, and as such there exist a large number of feature detectors (see e.g. [16] for a survey). Common among most of them is that they try to find corner-like features in the images; flat areas with uniform brightness are not distinctive enough to match unambiguously across images, and the same is true for edges, see Fig. 2.1.

Corner detectors work by computing some statistics of the extracted image patch. For example, they might examine the Hessian, the Laplacian-of-Gaussian [17], or the Difference-of-Gaussian (DoG) [18] of the image. Other detectors examine the auto-correlation function of the image [19, 20]: imagine extracting a rectangular patch centered around the point. If the contents of the patch changes considerably as we slide the window in any direction (with differences measured in, perhaps, sum-of-squared difference in pixel intensity between the original patch and the translated one), then the point corresponds to a corner. On the other hand, if the patch contents only change in one direction, but are more or less constant in the other direction, the point likely lies on an edge, and it seems unlikely we would be able to accurately identify it in a different image of the same scene.

Figure 2.1: Three example image patches from an image. A feature detector should trigger on corner-like points we believe we could re-identify in a different image. Do you think you could find and correctly match the three extracted patches in a different image of the same scene?

2.2 Feature descriptors

After having identified the keypoints in an image by running a feature detector on it, we need some way to encode the appearance of the keypoint, so that we can match it across images, or match it to a corresponding 3D point in a point cloud. While it is certainly possible in some cases to use a very simple similarity metric between image patches such as correlation or sum-of-squared-differences (SSD), the perhaps most widely used way of encoding image patch appearance is the gradient histogram [21]. The very popular SIFT and SURF features [22, 23] are examples of features which utilize a gradient histogram for describing the keypoints.

To compute the gradient histogram of a patch, the gradient (horizontal and vertical derivatives of the pixel intensity) are first computed for each pixel. The gradients are then binned into eight different bins, depending on their direction. I.e., the sum of the lengths of all gradients pointing in a direction between 0° and 45° are put in the first bin, the sum of lengths of all gradients pointing in a direction between 45° and 90° are put in the next bin, etc. This yields a total of eight numbers. An image patch can thus be compressed into eight numbers, representing, in some sense, in which directions any edges are oriented, and how strong these are.

However, when condensing a patch centered on a keypoint, the patch is first subdivided into 4×4 sub-patches, and each of these sub-patches is compressed into a vector of eight numbers using the above method. All

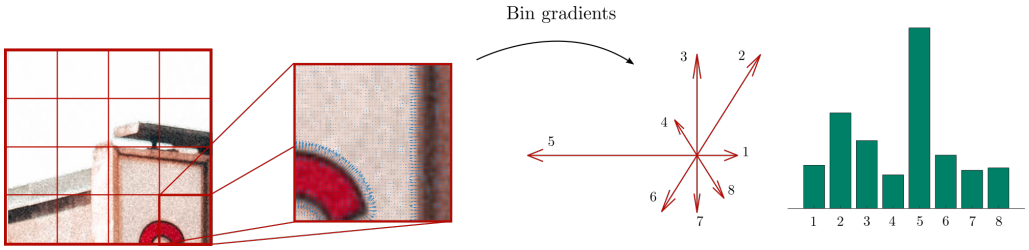


Figure 2.2: Illustration of gradient-histogram computation. The image is divided into 4×4 sub-patches. For each sub-patch, gradients are computed at each pixel and their magnitudes are added into one of eight bins depending on their orientation. The total values in these bins yield a descriptor-vector with eight entries.

these 16 vectors are then stacked into a single vector of 128 numbers. This vector is then the SIFT-descriptor of the patch. This process is illustrated in Fig. 2.2 for one of the image patches from Fig. 1.2.

Now, in order to find point correspondences between two images, or an image and a map, these local features (consisting of both the set of detections returned by the feature detector, as well as the corresponding feature descriptors) can be matched by, for each feature in the image, finding the nearest neighbour in the descriptor space in the other image (or map).

The SIFT descriptor, while simple, is remarkably effective at matching features across images, even under moderate changes of illumination and perspective distortions. While more involved descriptors have been proposed, such as descriptors learned from data using deep neural networks, the SIFT feature has turned out to be surprisingly difficult to consistently beat: it performs well on images from a broad range of categories, while learned descriptors typically performs well in a narrow range of images which resemble images or scenarios it has seen during training. SIFT currently remains one of the most popular baselines for comparison when developing new image features.

2.3 Feature matching across large appearance changes

While very powerful, the SIFT feature is not without its limitations. It struggles to find correct matches between images taken from very different viewpoints, such as between two images showing the same street intersection but taken from perpendicular streets, or between two images of the same scene taken during dissimilar environmental conditions. For example, reliably establishing feature matches between daytime and nighttime images, or between images taken during different seasons (see Fig. 2.3), remains

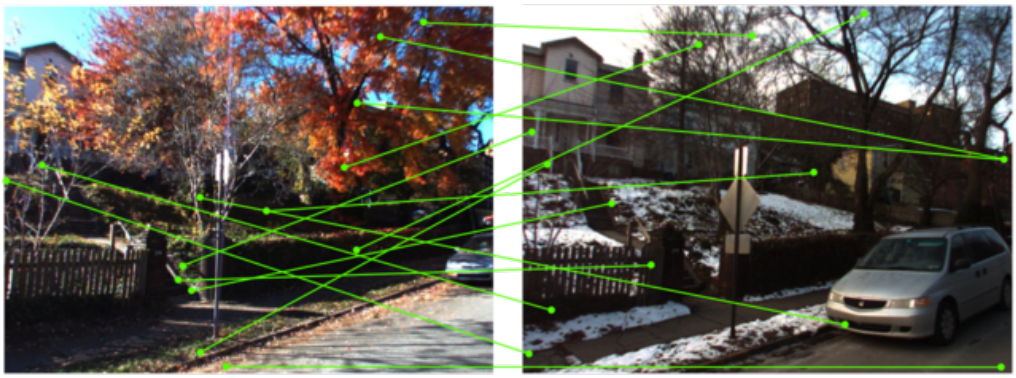


Figure 2.3: An example of SIFT-feature matching between two images taken from approximately the same viewpoint, but during different seasons. A Lowe ratio of 0.9 was used in the matching process. Not all feature matches are shown, but only those consistent with the estimated epipolar geometry.

very challenging and is still an open problem, and this is also why long-term visual localization is difficult. This should come as no surprise, since the SIFT descriptor is based only on low-level photometric intensity and gradient information: no high-level understanding or reasoning is involved during the matching process.

Fig. 2.3 shows an example of SIFT-feature matching between two images taken from approximately the same viewpoint, but far apart in time (during different seasons). Keypoints are extracted using the DoG detector, and then each feature in the left image is matched using approximate nearest neighbour matching (in descriptor space) to the features in the image to the right. A relative camera geometry which is consistent with as many matches as possible is calculated, and the figure shows the surviving, geometrically verified matches. None of them are correct.

The reason for this failure is two-fold: firstly the detector does not trigger on the same set of points in the two images (i.e., the detector is not *repeatable* over these appearance variations), and secondly the descriptor for a given point changes too much between the images to be reliably matched. The low-level pixel intensity in a local neighbourhood around any given patch looks completely different in the two images, even though they correspond to the same point. In the article [24], several extensive experiments are performed showing (and quantifying) the non-repeatability of the most popular feature detectors for varying degrees of viewpoint and lighting changes.

If a map is constructed from images captured during the same condition as in the left image, and we then revisit the same area at a later time, when the environment looks as in the figure to the right, how do we perform robust visual localization if traditional local-feature matching yields

mostly incorrect matches? This is the core problem which is discussed in the appended papers in this thesis.

At this stage, we can make one key observation regarding this simple experiment. If I asked you to manually provide a set of, say, ten point correspondences between the two images, would you be able to? Ideally, if they are to be used for camera pose estimation, the error when "clicking out" the correspondences should not be more than a few pixels.

Most likely you would. Though the low-level content of the image may have changed drastically in any given image patch due to changes in lighting, added snow, removed leaves and so on, we can (perhaps with some slight effort) match points based on a higher-level, semantic reasoning. We could match the corners of the building roof, the corners of the street sign, the tips of the poles in the picket fence, and so on. None of this would be possible by simply comparing the low-level image content between pairs of patches; we must reason using considerably more high-level information about the image contents to find the correct matches.

Machine learning based methods which extract this kind of high-level understanding from images have seen a very rapid increase in performance the past few years due to the advent of techniques for training deep neural networks (see [25] for a deep-learning review). Employing deep-learning techniques to extract semantic meaning from images for localization purposes, or to robustly identify and match semantic keypoints between images even under these challenging conditions thus seems like a promising avenue for research. Though there are many works in the literature attempting to learn local features (e.g. [26–28]), no learned method has yet emerged which seems to consistently outperform SIFT for general images.

3 Structure-based localization

Now that we have understood how to identify local features in images, and how these can be matched between images, we will now have a look at how these features can be used for the task of visual localization. We will first have a look at structure-based methods, i.e., methods which employ a 3D representation of the environment (such as a point cloud) to aid in the localization process. In order to understand how this is used, we must first have a quick look at the image formation process for cameras, i.e., the mechanism by which the 3D world is projected down into a two-dimensional image.

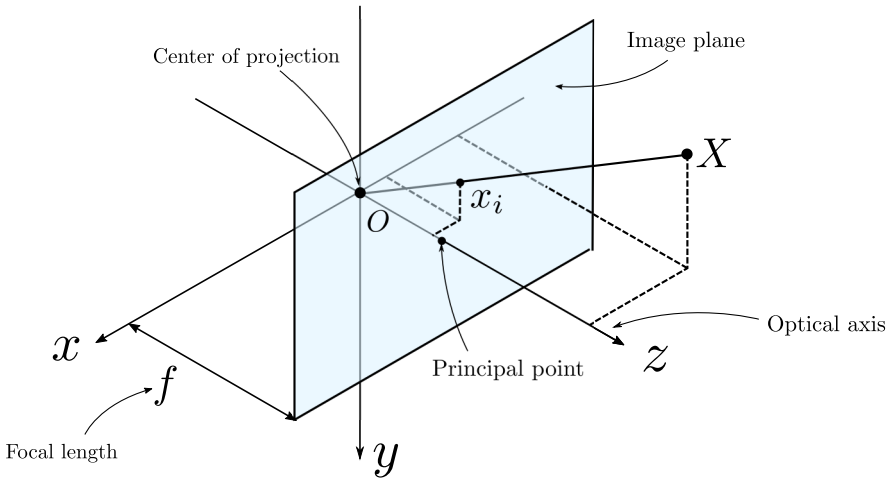


Figure 2.4: Projection of a 3D point X onto the image plane of a pinhole camera.

3.1 The camera model

The standard pinhole camera model is a subject which has been written extensively about before, and every course on optics, computer vision and computer graphics covers this, so it seems somewhat redundant to write yet another introduction. We will here only write down the very basics of camera projection. For more information, see e.g. Chapter 6 in [29].

The simplest and perhaps most popular camera model is the pinhole camera model. A pinhole camera, or a *camera obscura*, is simply a box with a tiny hole (a pinhole) in one of its sides. When light is emitted from, or scatters off of, an object in the scene, some of that light might pass through the hole, which we will call the center of projection O, and will fall onto a point on the opposite side of the box. An image of the scene will thus form on the inside of the box. If we were to place a photographic plate on the side opposite from O, an image would form on the plate. We have thus created a simple camera.

Fig. 2.4 shows the projection process for a point X, with one slight modification: the image plane where the image forms is now imagined to be in front of the center of projection. This is of course not what happens in practice; the image is formed on a plane a distance f behind the center of projection. However, the image which forms on the physical imaging plane will be a mirror image of what an observer would see when "looking out" from O. When displaying the captured image, it will have to be mirrored in order to display what was actually seen by the camera. To simplify the calculations, it is more convenient to imagine the image being formed on the *image plane* placed a distance f in front of the center of projection.

The relationship between the world point X and the imaged point x_i can now be deduced from similar triangles. Let $X = (X_w, Y_w, Z_w)$ and $x_i = (x, y, f)$, then clearly $x = X_w \cdot f/Z_w$, $y = Y_w \cdot f/Z_w$.

The equations for camera projections are most conveniently expressed in the framework of projective geometry. In this framework, it is customary to express points in 2D and 3D in terms of their homogeneous coordinates: the point (x, y) in $\mathbb{R}^2$ is now represented as $(x, y, 1)$. Furthermore, we identify all points along the same ray through the origin, i.e., the point (x, y) in Euclidean coordinates is identified with all points $\vec{x} = \lambda(x, y, 1)$ in homogeneous coordinates, with $\lambda \neq 0$. Similarly, the point $\vec{X} = (X_w, Y_w, Z_w)$ in $\mathbb{R}^3$ is identified with all points with homogeneous coordinates $\lambda(X_w, Y_w, Z_w, 1)$.

Using this formalism, the camera projection equations derived above can be written as

$$\lambda\vec{x} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I_{3\times3} | & \vec{0} \end{bmatrix} \vec{X}, \tag{2.1}$$

where $\vec{x}$ and $\vec{X}$ are the homogeneous representations of the points x and X, f is the focal length, $I_{3\times3}$ is the 3×3 identity matrix, and $\vec{0}$ is the vector of all zeroes in $\mathbb{R}^3$.

To obtain the final projection equations for the pinhole camera, two additional things need to be modified. First, the vector $\vec{x}$ is measured in meters (or whichever unit of length is used to express $\vec{X}$). These coordinates are often called the normalized image coordinates. However, when working with images, one initially obtains the image features in terms of their pixel coordinates in the image, counting rows from up to down, and columns left to right, with the pixel $(1, 1)$ being in the top left corner. To transition from normalized coordinates to pixel coordinates, one multiplies the normalized coordinates by the camera calibration matrix K given by

$$K = \begin{bmatrix} \alpha_x & s & c_x \\ 0 & \alpha_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.2}$$

where α_x and α_y is the focal length in terms of pixels (these are identical if the pixels are square), s is called the skew and is zero for rectangular pixels, and the point (c_x, c_y) is the principal point in pixel coordinates, i.e., the point where the optical axis intersects the image plane.

Lastly, the camera may not actually be located at the origin in coordinat system of the 3D world, and it need not be oriented as shown in Fig. 2.4. Suppose instead that the center of projection is located at some arbitrary position $\vec{C}$, and that the camera has some orientation described by the

rotation matrix R, which brings the camera coordinate axes to the world coordinate axes, then the final projection equation may be written

$$\lambda\vec{x} = P\vec{X}, \tag{2.3}$$

where

$$P = KR[I_{3\times3}| - \vec{C}] \tag{2.4}$$

is called the camera matrix, which fully determines the projection of world points into image coordinates.

Note that the camera matrix contains five intrinsic parameters, related to the camera's internal workings (sensor size, focal length, skew, etc.), and six external parameters (related to the position of the camera in the external world), for a total of eleven parameters.

In the visual localization problems discussed in the papers in this thesis, the internal parameters have all been determined beforehand in a calibration procedure. This is typically done by taking several pictures of a calibration object whose physical dimensions are known very precisely [30].

One final thing worth bringing up before moving on to structure-based camera pose estimation, is that the pinhole camera model is an idealized, mathematical model for the image formation process. In practice, since the pinhole camera only lets in a very small amount of light, a larger aperture is needed (the exposure time for the world's first pinhole camera was around eight hours [31]). However, increasing the aperture size leads to blurry images in a pinhole camera, so a lens is placed in the aperture to focus the light. Real optical systems can be quite sophisticated, and the lenses introduce several different kinds of aberrations in the optical system, one of which is that of non-linear distortion.

An ideal pinhole camera maps lines in 3D to lines in the image, but non-linear distortion causes these lines to map onto curved arcs in the image, often with increasing distortion as one moves away from the principal point. However, this type of distortion has been studied for a long time, with its roots in the photogrammetric community [32, 33], and simple and accurate models for the non-linear distortion have been created. These distortion parameters are typically estimated during the camera calibration procedure [30], and distorted images can then be *rectified* into corresponding pinhole camera images with sub-pixel accuracy.

3.2 Correspondence-based camera pose estimation

Now that we understand the camera model, we are in a position to look at how structure-based pose estimation works. The problem of estimating the camera pose from a set of point or line correspondences is perhaps one of

the oldest problems of computer vision, and there exists a vast literature on the subject. In this section we will summarize the most important points needed to understand the papers in Part II.

Minimal solvers

Assume that we have identified as set of points $\{x_i\}_{i=1}^n$ in the image, and have matched these to a set of 3D points $\{X_i\}_{i=1}^n$ in a 3D point cloud. We can then write down the projection equation 2.3 for each 2D-3D correspondence. Note that the projection equations yield three equations per correspondence, but each correspondence also introduces an extra unknown λ. Each correspondence thus fixes a net two degrees of freedom in the camera matrix. If the camera intrinsics are known, e.g., if we have calibrated the camera beforehand, there are a total of six unknown degrees-of-freedom in the camera matrix, corresponding to the position of the camera center, and the camera orientation.

It thus seems a total of three correspondences should in general suffice to compute the camera pose. This is indeed the case. This problem is known as the perspective-three-point problem (P3P), and many different methods for solving this problem have been presented over the years [6, 7, 34], with the first known solution dating back to 1841 [35].

A classical way of solving the problem is to, for each pair of points, use the law of cosines to formulate a quadratic equation in the unknown depths of the 3D points [8, 36, 37]. The three possible pairs thus yield three quadratic equations in the three unknown depths. One may solve this system in a variety of ways. One way is to use the Sylvester resultant to successively reduce the system into a single fourth degree equation, which will yield up to four possible solutions to the problem [34]. In general, a fourth correspondence is needed to disambiguate the solution. For four or more point correspondences, there exists linear methods for calibrated camera pose estimation [38, 39].

In computer vision, methods which solve a problem using only the minimum number of correspondences required theoretically are often called *minimal solvers*. They play a central role in robust estimation, which we will see below.

The P3P solvers mentioned above is one kind of minimal solver, but there exists other kinds of minimal solvers for camera pose depending on what kind of information is available. For example, if the gravity direction is known in the camera reference frame (perhaps supplied by an IMU, or estimated from vanishing lines in the image [40]), the camera pose only has four extrinsic degrees of freedom left, enabling pose estimation from only two point correspondences [41].

Robust estimation

Once correspondences between the image and a map have been established, inputting all correspondences in an n-point pose solver to estimate the camera pose is a very bad idea. The reason is the presence of *outliers*, in the data, i.e., mismatched correspondences which are completely wrong. Including these in a pose estimation routine which minimizes an algebraic error or the geometric reprojection error of the correspondences will introduce gross errors in the estimated pose. A way to remove outliers, or to downweight their contribution to the error function must be devised.

The undoubtedly most popular technique for outlier detection and removal in computer vision is the *random sample consensus maximization* (RANSAC) procedure [8]. This method is not limited to only camera pose estimation, but works in a wide variety of robust estimation problems, such as line and plane fitting, 2D and 3D homography estimation, and so on.

The main idea is to randomly sample a minimal subset of the correspondences, i.e., extract a subset which is just large enough to apply a minimal solver on it. For calibrated camera pose estimation, a subset of three correspondences would be extracted. Once the camera pose (or homography, or whichever type of model it is we are estimating) has been computed from the minimal subset, we check how many of the remaining correspondences agree with this model, within some pre-specified error tolerance. For camera pose estimation, we may project down the 3D points of the remaining correspondences, and measure the reprojection error. We count the number of inliers (the size of the *consensus set*) for the estimated model. We repeat this procedure a pre-specified number of times, and simply return the model with the largest consensus set.

Typically, the final pose is then refined using the whole consensus set, using an iterative local optimization procedure to minimize the reprojection error of the correspondences in the consensus set.

RANSAC is not the only possible approach to handling outliers, and there exists a large body of work concerned with developing other kinds of robust estimation techniques, such as globally optimal model fitting under the L_2 norm [42], and methods based on branch-and-bound [43, 44], however, these methods tend to suffer from an exponential worst-case runtime.

Due to its simplicity and generality, RANSAC remains the most popular robust estimation method. However, it has several shortcomings worth mentioning. First, RANSAC is not clearly not guaranteed to return the optimal solution. Secondly, the algorithm may be sensitive to the selected error threshold: the model calculated from a set of correct correspondences may not actually have a large consensus set due to noisy observations. Lastly, the runtime of RANSAC is exponential in the ratio of outliers. This is es-

pecially a problem in long-term visual localization, where a large fraction of the correspondences are expected to be incorrect due to the large appearance variations of the scene, as well as in large-scale localization, where the number of correct matches typically decreases with model size due to visual ambiguity resulting from repetitive structures in the environment [45] (i.e., the corners of a window in the observed image may look very similar to windows on a building on the other side of the city).

Due to the exponential dependence on the outlier ratio, it is sometimes desired to prune outliers if possible before feeding the correspondences into the RANSAC procedure. This leads us to a class of methods often called *outlier filters*, or *outlier rejection schemes*.

Outlier rejection

If a large amount of outliers are expected, it may be beneficial to utilize an outlier rejection scheme. Typically, these use prior information about the camera pose to reason about which correspondences may be outliers. For example, [46] presents an outlier rejection scheme where the camera rotation is fully known; in this scenario it is possible to, for each correspondence, calculate an upper bound on the number of inliers possible for a camera pose which also has that particular correspondence is an inlier. If this score is low, it can then be discarded. The article [47] presents an outlier rejection scheme which utilizes prior information about camera height and vertical direction. One of the included papers in this thesis presents an outlier rejection scheme based on exploiting the semantics of the observed image.

4 Image-retrieval based localization

In the beginning of the chapter, we discussed that visual localization methods can be roughly categorized into metric and topological. The metric methods are typically 3D structure based, i.e., they employ a 3D model of the environment and use this to calculate the pose of the camera which captured the query image, as discussed in the previous section.

The topological localization methods, also called *visual place recognition* methods, work in a different manner. Here, the localization problem is instead formulated as an image search problem: given a set of database images and a query image, find the database image which most closely resembles the query image, see Fig. 2.5.

If metric information is included in the map, for example if the images are geotagged and have associated GPS metadata, then the position of the query image could be approximated by the position of the database image

Figure 2.5: Visual place recognition formulates the localization problem as an image retrieval problem. Given a query image and a set of database images, find the database image most similar to the query image (according to some metric).

[48].

One of the advantages of image retrieval based approaches is that image retrieval is a well-studied problem in computer vision, and efficient search strategies using vocabulary trees [49] and inverted file indices have been developed for this problem.

A common implementation of image retrieval is to compute a whole-image descriptor for all database image. The whole-image descriptor could be based on a bag-of-visual-words (BoW) approach [50–53], or use a vector-of-aggregated-descriptors (VLAD) [54, 55]. The basic idea behind these methods is to extract local features from the image, such as SIFT descriptors. If the descriptor space has been quantized beforehand, for example by clustering all descriptors from a different set of training images using a k-nearest-neighbour approach, each descriptor extracted from the query image can be assigned to the nearest cluster. The corresponding whole-image descriptor would then be the histogram over the number of features assigned to the different clusters (visual words).

To localize an image, this whole-image descriptor is computed for all database images, and for the query image, and the nearest-neighbour (or k-nearest neighbours), are extracted from the database. The pose of the query image can then be approximated using the pose of the top-ranked database image.

Computing a global descriptor from extracted local descriptors yields a

more compact representation of the image, but spatial information is lost. To compensate for this, spatial verification and re-ranking [56] can improve the results. This means that the top-ranked images after image retrieval are then re-ranked based on how well it is possible to fit some geometric transformation that maps correspondences from the query image to the database images in questions. In other word, this approach tries to take the spatial layout of the features into account to reconsider the ordering of the suggested best matches. For a survey of the visual place recognition problem, see [57].

Image retrieval methods have on significant advantage over structure-based methods: a database of geotagged images is significantly easier to construct, maintain and extend than a metric 3D reconstruction. It was believed that structure-based approaches yielded more accurate pose estimates than image retrieval based methods, but there is recent evidence that estimating the camera pose using a local 3D model created "on-the-fly" from the top-ranked database images can yield just as accurate, if not more accurate, poses than pure structure-based methods [58].

5 Semantic segmentation

When discussing local image features in Section 2, we discussed the problem of non-invariance of local features. That is, under viewpoint and illumination changes, the feature detector may trigger on a different set of points, and the feature descriptor may change to such an extent that feature matching based on descriptor distances fails completely.

The problem is that the descriptors only contain very low-level intensity-gradient information in a patch around the feature points. They contain no higher-level understanding of the scene. Even in a fairly challenging scenario, a human would likely be able to provide matches between two images by utilizing this high-level information.

In the computer vision community, much progress has been made in recent years on the problem of semantic segmentation. This problem consists of assigning, to each pixel in an image, a label from a pre-defined set of classes. Which classes are used depends on the problem at hand. Fig. 2.6 shows an example of an image together with its semantic segmentation. The image comes from the CMU Visual Localization dataset [10] and the segmentation is performed using the network [59] trained on the Cityscapes classes [60, 61]. Cityscapes is a dataset and public benchmark for semantic segmentation of street-view images into semantic classes such as road, sidewalk, car, pedestrian, pole, building, sky and so on. These aim towards a high-level understanding of images taken in street scenes, which may be

Figure 2.6: An image from the CMU Visual Localization dataset and its corresponding semantic segmentation.

relevant for the task of visual navigation for autonomous cars.

Today, semantic segmentation is most commonly performed using some variants of convolutional neural networks (CNNs) [62–64]. Sometimes, a conditional random field (CRF) model is added on top of the network output to encourage a structured segmentation [64–66].

Even though the local image features may not be invariant to illumination, seasons, day-night changes and so on, a good semantic segmentation algorithm should ideally be robust to these kinds of variations. If semantic information can be reliably extracted under these conditions, it may be possible to utilize this during the localization process. For example, it may be used for identifying incorrect matches (a feature detected on a street sign should be matched to a 3D point with the corresponding label), as done in e.g. [67] in the context of object retrieval.

As observed in [67], the idea of integrating semantic information into the classical computer vision pipelines is something of an emerging theme in the computer vision literature, since this has been shown to improve the performance across several different tasks. For example, in [68], the authors show that for the problem of dense stereo reconstruction in road scenes, jointly reasoning about depth and semantic classes improves the performance of the reconstruction dramatically. The article [69] reaches similar conclusions, where they instead jointly infer semantic labels and depths for ”stixels”. Similarly, in [70] a voxelized multi-view-stereo reconstruction is performed by joint geometric and semantic reasoning. It is found that the geometry helps enforce the semantics across images, and the semantics help the depth reconstruction e.g. in areas (such as the ground) for which depth values are sampled more sparsely.

5.1 Semantics for visual localization

Lastly, to wrap up this chapter, we arrive at semantics for visual localization, which is one of the key topics of this thesis. As mentioned above, integrating semantics into the classical, geometrical pipelines has been shown to improve the performance. The question is then whether this is also the case for visual localization. There exists evidence this is indeed the case (see for example [71–73] for examples where higher-level features such as lane-markings and pole-structures are used for localization), and it is also the topic of three of the articles appended in Part II.

Chapter 3

Thesis Contributions

The topic of this thesis is that of long-term visual localization. Current visual localization approaches typically work well when the mapping images are similar in appearance and view-point to the images to be localized, but under change of viewpoint, illumination or seasons, the localization performance typically degrades rapidly. The non-repeatability of the feature detector, and the non-invariance of the feature descriptor are the two main culprits.

We have mentioned in the last chapter that the reason behind the failure of the local feature approach in this scenario is that they rely only on low-level pixel-intensity information, and that incorporating a higher-level scene understanding via the semantic segmentation may potentially alleviate some of these problems. This is precisely the topic of three of the included papers (Paper II, III, and IV).

While working on the first two articles (i.e., articles III and IV), we noticed the lack of suitable long-term visual localization datasets to evaluate our methods on. There were a variety of localization datasets available, but they either did not have sufficient variation in weather, seasons or illumination to evaluate the performance of the methods in the long-term localization scenario, or, if they did include this variation, they did not come with accurate six degree-of-freedom camera poses. Motivated by this, we joined forces with a quite large group of researchers and put together three datasets (where we at Chalmers were responsible mainly for one) specifically aimed at evaluating six degree-of-freedom long-term visual localization. This work resulted in the most recent article, Paper I.

In the remainder of this chapter we present a high-level overview of each of the included papers in turn. In Part II, the papers are presented in reverse-chronological order starting with the newest and ending with the oldest. Here, we instead present the work in the original order in which it was produced, starting with the earliest and ending with the most recent.

Figure 3.1: Illustration of the method in Paper IV. Any given pose may be evaluated by how well the semantically labelled structure projects down into the given pose. Only the semantic 3D curves are shown (poles, road edge, vegetation-sky contour). 3D points (not shown) were used as well.

1 Paper IV

C. Toft, C. Olsson, F. Kahl. "Long-term 3D Localization and Pose from Semantic Labellings". *3D Reconstruction Meets Semantics Workshop at the International Conference on Computer Vision 2017.*

This paper was something of a pilot-study where we examined how well it is possible to perform single-image visual localization, using only the semantic segmentation of the query image. In other words, is the semantics alone sufficient for visual localization? The method was evaluated on two small subsets of the Oxford RobotCar dataset, each of the size of a few city blocks. The method uses an image-retrieval method (based only on the semantic segmentation), and then refines the 6 DoF camera pose by minimizing a cost function based on how well the reprojection of a semantically labelled 3D model consisting of semantically labelled points and curves of the environment matches the observed segmentation in the image. The results suggest that semantics is sufficient for localization in small environments when the query image is sufficiently "semantically interesting", and that the proposed 6 DoF semantic pose refinement does indeed improve the pose.

Author contribution. I did most of the implementation work of the method and the evaluation on the datasets. The third author created the datasets used and contributed the original ideas. The second author con-

tributed to the discussions and assisted greatly in writing the article.

2 Paper III

E. Stenborg, C. Toft, L. Hammarstrand. "Long-term Visual Localization using Semantically Segmented Images". *International Conference on Robotics and Automation 2018.*

This article is something of an improvement or extension of the methods and ideas presented in Paper IV, but applied to the problem of sequential localization using recursive Bayesian estimation methods. That is, the input to the method is a whole sequence of images and inertial measurement data, and the goal is to calculate the most likely position of the vehicle at the current time, given all the previous measurements up to the present. The basic idea is very similar to that in paper IV, where a tentative pose can be evaluated by how well the reprojected 3D structure agrees with the observed semantic labellings of the image. However, in this article, this idea is developed in a probabilistic particle filtering framework, where the measurement likelihood function is defined by this semantic consistency between the observed semantic labellings, and the reprojected semantic 3D structure. The advantage of this approach is that it allows inertial measurement data to be naturally incorporated into the localization pipeline.

Author contribution. The main ideas of how to formulate the problem in a Bayesian estimation framework were developed by the first and third authors. Implementation of the semantic localization was done by the first author. I was mainly responsible for implementing the reference solution and assisted in the writing process.

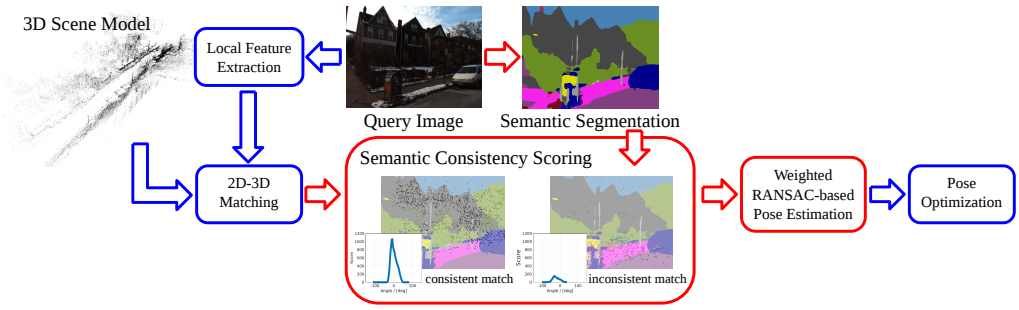


Figure 3.2: The general pipeline of the method presented in paper II. The method is based on a traditional feature-based pipeline, but each feature is scored depending on how well it agrees with the overall semantics of the scene and image. This score is then used to weight the sampling probabilities in the RANSAC loop.

3 Paper II

C. Toft, E. Stenborg, L. Hammarstrand, L. Brynte, M. Pollefeys, T. Sattler, F. Kahl. "Semantic Match Consistency for Long-Term Visual Localization". *European Conference on Computer Vision 2018.*

This paper presents another semantics-based single-image localization method. Unlike papers III and IV, which were based on semantics alone, this article aims to increase localization performance by incorporating semantic information in a classical geometrical pose-estimation pipeline based on image features.

Specifically, we address the problem of the high-outlier ratios which are encountered in feature-based methods in the long-term localization scenario. Inspired by the geometric outlier-filtering methods, we devise a semantic outlier filter, which, like the geometric outlier filters, use prior knowledge about camera height above ground plane as well as vertical direction, to reason about the likelihood of each correspondence being an inlier. The method is evaluated on two self-driving car datasets, and we show that, under the conditions mentioned, the method can significantly increase the performance of a classical P3P RANSAC based pipeline.

One of the main advantages of the proposed method is that it is fully parallel in the number of correspondences: each correspondence can be scored independently of the others, and the method of scoring correspondences depends only on projections and angle calculations. As such, it would be very suitable for an efficient GPU implementation, though in the paper only a sequential MATLAB implementation was performed.

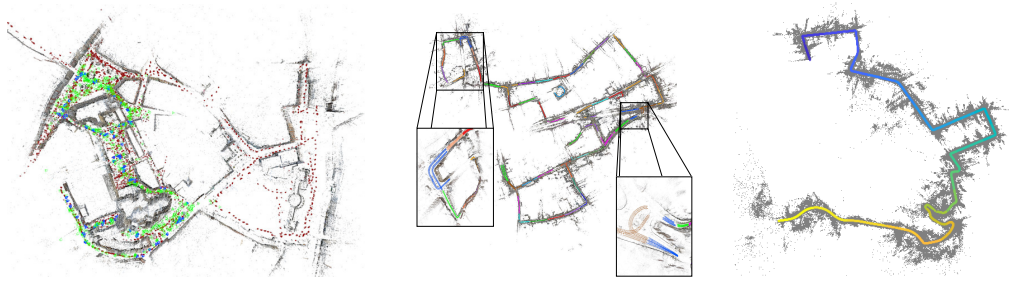


Figure 3.3: Maps of the three new datasets we present in Paper I: the *Aachen Day-Night dataset* (left), the *RobotCar Seasons dataset* (middle) and the *CMU Seasons dataset* (right).

Author contribution. The last author supplied the original idea, and I implemented the method and all evaluation scripts. The other authors helped very much with discussions, writing, figures and segmentations.

4 Paper I

C. Toft, T. Sattler, W. Maddern, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, T. Pajdla and F. Kahl. "Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions". *To be Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence 2019.*

This is a dataset paper, where we present three challenging new datasets for visual localization. In this article, we address our observation that there were no public datasets suitable for evaluation of six degree-of-freedom visual localization in the long-term scenario. There were several datasets containing images taken under a variety of different conditions (such as several traversals of the same road in winter, summer, spring, day, night etc.), however these did not come with known reference poses, making them unsuitable for evaluation of long-term localization algorithms.

In this article, we augmented three of these publicly available datasets with reference poses by reconstructing each condition individually, and then registering the different models into the same coordinate system. The dataset we augmented were the CMU Visual localization dataset [9], the Aachen dataset [74] and the Oxford RobotCar dataset [75].

The article also presents an evaluation of the current state-of-the-art methods on these datasets, such that other researchers will not have to spend time evaluating baseline methods.

Author contribution. This work was a large collaboration between many researchers and research groups. At Chalmers, we were mainly responsible for the CMU Seasons dataset. Fredrik did most of the bundle adjustment and establishing ground truth poses. I triangulated the reference models, ran some of the baseline methods, and formatted and organized the raw dataset into its current, published form, and was responsible for putting together the journal article and setting up the benchmarking server and website. Torsten did most of the writing in the original CVPR article.

Chapter 4

Conclusion and Future Outlook

In this thesis, we have presented work on the problem of robust long-term visual localization mainly by adressing two points:

- Several possible ways to incorporate semantic information to aid in the visual localization process.
- Develop new datasets to enable accurate and fair evaluation of the performance of long-term localization methods.

In terms of incorporating semantic information into the localization process, the appended papers show some encouraging results: it seems that semantic information does indeed contain very valuable information for pose estimation, and it seems this can be used to complement the traditional geometrical pipelines to improve the results, or obtain results on par with the geometric pipelines. But the story is far from over.

There are many shortcomings in the papers which should be addressed in order for them to approach something of more practical utility. At the moment, one of them is speed: the presented methods (or at least, the current implementations of them) are too slow to be used in a real-time system for visual navigation of a real vehicle, or to be used in a real-time virtual reality application. While there are of course more applications to camera pose estimation than autonomous navigation and virtual reality, this is the scenario the datasets used (and created) have been aimed at.

A related issue is that of a lack of accuracy requirements. A question which has recurred continually throughout the work has been that of: *What localization error is acceptable?* How do we know when the localization problem has been solved? What framerate do we need to be able to handle? At the moment, I am not aware of any work which provides general answers to these questions. The localization performance of the state-of-the-art algorithms on our new benchmark datasets suggest that current methods

are unable to handle drastic visual changes satisfactorily, but it is not clear what accuracy is desired, or realistic to expect.

Another question is what formulation to use for the visual localization problem. Since the used datasets are aimed towards an autonomous driving scenario, it seems artificial to focus only on the single-image localization scenario when an entire history of image and IMU data is readily available. This would also somewhat lessen the requirements on the single-image localization performance: it would not necessarily be catastrophical if the visual localization fails if we already have a good idea of where we are and where we are going. Indeed, requiring all images to be localized is unrealistic. In a real street scenario, it is unavoidable that some images will be impossible to localize on their own. They might be overexposed due to the sun hitting the camera straight on, the entire image might be covered by a large truck driving by, or we might just see uninformative vegetation, such as a bush covering the entire image. As such, I believe the sequential localization formulation is more natural in the context of visual navigation of autonomous cars.

1 Future work

I believe there are several interesting directions for future work in terms of long-term visual localization, and I will outline some of them below.

1.1 Learned features

One approach, which has been gaining in popularity, is to try to learn a feature detector and descriptor. In other words, train a detector which tends to trigger on points which are long-term stable, i.e., points which reoccur (and can be re-identified) even after a significant period of time has passed. This might be corners of windows, rooftop corners, street signs, and so on (or any other kind of point which the system can reliably re-identify). One of the problems which may arise when training this form of feature, is that ground-truth correspondences will be needed, i.e., the system needs to know which features actually are stable such that it can learn from these. We hope our new datasets will be useful to other researchers in this regard.

1.2 Incorporate 3D information during feature matching

Another approach is to use features based on 3D information, or to incorporate 3D information in the feature matching process. There has been

some recent work which obtains promising result by utilizing semantic 3D descriptors [76]. That paper constructed a voxel model of the environment. Perhaps something similar could be done for point clouds. One might then extract depth information from the query image, using e.g. multi-view-stereo or monocular depth-estimation, perhaps together with estimated surface normals and semantics, to obtain a more discriminative descriptor of the feature point. The feature descriptor would then not only encode the 2D appearance of the point in the image, but also information about the point and its surroundings in the 3D world.

1.3 Better semantic segmentations

Lastly, one problem we encountered when trying to exploit the semantic segmentations for long-term visual localization, was that the semantic segmenters themselves were not particularly robust to these kinds of appearance and viewpoint changes. It turns out that most segmenters are trained on street images taken during daytime in fairly nice weather conditions, with a camera mounted on the car such that it faces the road "straight on". If the camera is mounted on the side of the car such that it is facing left or right, or if the weather is different (during nighttime, with snow, with golden sunlight from the autumn sun and so on), the segmentations turned out less than ideal.

If we wish to use semantics for long-term visual localization, then the semantic segmenter itself must also be robust to these kinds of changes. Improving semantic segmentations over challenging conditions is thus a desirable goal with an immediate synergy with visual localization.

Bibliography

[1] N. Snavely, S. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3d", *ACM SIGGRAPH*, 2006.

[2] J. L. Schönberger and J.-M. Frahm, "Structure-From-Motion Revisited", in *CVPR*, 2016.

[3] E. Kaplan and C. Hegarty, *Understanding GPS: Principles and Applications, Second Edition*, ser. Artech House mobile communications series. Artech House, 2005.

[4] D. Lowe, "Distinctive image features from scale-invariant keypoints", *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[5] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration", in *Int. Conf. on Computer Vision Theory and Applications*, 2009.

[6] T. Ke and S. I. Roumeliotis, "An efficient algebraic solution to the perspective-three-point problem", *arXiv preprint arXiv:1701.08237*, 2017.

[7] L. Kneip, D. Scaramuzza, and R. Siegwart, "A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation", in *CVPR*, 2011.

[8] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Communications of the ACM*, 1981.

[9] H. Badino, D. Huber, and T. Kanade, "Visual topometric localization", in *Intelligent Vehicles Symposium (IV)*, Baden-Baden, Germany, Jun. 2011.

[10] H. Badino, D. Huber, and T. Kanade, "Visual topometric localization", in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, IEEE, 2011, pp. 794–799.

[11] D. Xu, H. Badino, and D. Huber, "Topometric localization on a road network", in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, IEEE, 2014, pp. 3448–3455.

[12] H. Badino *et al.*, "Real-Time Topometric Localization", in *ICRA*, 2012.

[13] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization", in *International Conference on Computer Vision*, 2015.

[14] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer-Verlag, 2010.

[15] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Second. Cambridge University Press, ISBN: 0521540518, 2004.

[16] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors: A survey", *Foundations and Trends in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, 2008.

[17] T. Lindeberg, "Feature detection with automatic scale selection", *International journal of computer vision*, vol. 30, no. 2, pp. 79–116, 1998.

[18] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[19] C. Harris and M. Stephens, "A combined corner and edge detector", in *Alvey Vision Conference*, 1988.

[20] B. Triggs, "Detecting keypoints with stable position, orientation, and scale under illumination changes", in *European Conference on Computer Vision*, vol. 4, Prague, Czech, 2004, pp. 100–113.

[21] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection", in *CVPR*, San Diego, USA, 2005, pp. 886–893.

[22] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[23] H. Bay *et al.*, "SURF: Speeded up robust features", *European Conference on Computer Vision*, vol. 110, no. 3, pp. 346–359, 2008.

[24] H. Aanæs, A. L. Dahl, and K. Steenstrup Pedersen, "Interesting interest points", *International Journal of Computer Vision*, vol. 97, no. 1, pp. 18–35, Mar. 2012.

[25] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", *nature*, vol. 521, no. 7553, p. 436, 2015.

[26] K. M. Yi *et al.*, "Lift: Learned invariant feature transform", in *European Conference on Computer Vision*, Springer, 2016, pp. 467–483.

[27] Y. Ono *et al.*, "Lf-net: Learning local features from images", *CoRR*, vol. abs/1805.09662, 2018. arXiv: 1805.09662.

[28] N. Savinov *et al.*, "Quad-networks: Unsupervised learning to rank for interest point detection", in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[29] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[30] Z. Zhang, "A flexible new technique for camera calibration", *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, 2000.

[31] E. Hecht, *Optics*. Addison-Wesley, Reading, Mass., 1987.

[32] D. C. Brown, "Close-range camera calibration", *Photogramm. Eng*, vol. 37, no. 8, pp. 855–866, 1971.

[33] W. Faig, "Calibration of close-range photogrammetric systems: Mathematical formulation", *Photogrammetric engineering and remote sensing*, vol. 41, no. 12, 1975.

[34] X.-S. Gao *et al.*, "Complete solution classification for the perspective-three-point problem", *IEEE transactions on pattern analysis and machine intelligence*, vol. 25, no. 8, pp. 930–943, 2003.

[35] J. A. Grunert, "Das pothenot'sche problem in erweiterter gestalt; nebst bemerkungen über seine anwendung in der geodäsie", *Grunert Archiv der Mathematik und Physik*, 1841.

[36] E. Merritt, "Explicit three-point resection in space", *Photogrammetric Engineering*, vol. 15, no. 4, pp. 649–655, 1949.

[37] S. Linnainmaa, D. Harwood, and L. S. Davis, "Pose determination of a three-dimensional object using triangle pairs", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 5, pp. 634–647, 1988.

[38] A. Ansar and K. Daniilidis, "Linear pose estimation from points or lines", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 578–589, 2003.

[39] L. Quan and Z. Lan, "Linear $n \leq 4$-point camera pose determination", *PAMI*, vol. 21, no. 8, pp. 774–780, Aug. 1999.

[40] J.-C. Bazin *et al.*, "Globally optimal line clustering and vanishing point estimation in manhattan world", in *Conference Computer Vision and Pattern Recognition*, 2012.

[41] Z. Kukelova, M. Bujnak, and T. Pajdla, "Closed-form solutions to minimal absolute pose problems with known vertical direction", in *Asian Conference Computer Vision*, 2010.

[42] E. Ask, O. Enqvist, and F. Kahl, "Optimal geometric fitting under the truncated l_2-norm", in *Conference Computer Vision and Pattern Recognition*, 2013.

[43] O. Enqvist, K. Josephson, and F. Kahl, "Optimal correspondences from pairwise constraints", in *Computer Vision, 2009 IEEE 12th International Conference on*, IEEE, 2009, pp. 1295–1302.

[44] O. Enqvist and F. Kahl, "Robust optimal pose estimation", in *European Conference on Computer Vision*, 2008.

[45] B. Zeisl, T. Sattler, and M. Pollefeys, "Camera Pose Voting for Large-Scale Image-Based Localization", in *ICCV*, 2015.

[46] V. Larsson *et al.*, "Outlier rejection for absolute pose estimation with known orientation", in *British Machine Vision Conference*, 2016.

[47] L. Svärm *et al.*, "City-scale localization for cameras with known vertical direction", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1455–1461, 2017.

[48] G. Schindler, M. Brown, and R. Szeliski, "City-Scale Location Recognition", in *CVPR*, 2007.

[49] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree", in *CVPR*, vol. II, New York City, USA, 2006, pp. 2161–2168.

[50] O. Chum *et al.*, "Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval", in *International Conference on Computer Vision*, 2007.

[51] H. Jégou, M. Douze, and C. Schmid, "On the burstiness of visual elements", in *CVPR*, 2009.

[52] H. Jegou, H. Harzallah, and C. Schmid, "A contextual dissimilarity measure for accurate and efficient image search", in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, IEEE, 2007, pp. 1–8.

[53] A. Torii *et al.*, "Visual Place Recognition with Repetitive Structures", in *CVPR*, 2013.

[54] A. Torii *et al.*, "24/7 Place Recognition by View Synthesis", in *CVPR*, 2015.

[55] R. Arandjelovic and A. Zisserman, "All about vlad", in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2013, pp. 1578–1585.

[56] J. Philbin *et al.*, "Object Retrieval with Large Vocabularies and Fast Spatial Matching", in *Conference Computer Vision and Pattern Recognition*, 2007.

[57] S. Lowry *et al.*, "Visual place recognition: A survey", *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.

[58] T. Sattler *et al.*, "Are Large-Scale 3D Models Really Necessary for Accurate Visual Localization?", in *CVPR*, 2017.

[59] F. Yu and V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions", in *ICLR*, 2016.

[60] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding", in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[61] M. Cordts *et al.*, "The cityscapes dataset", in *CVPR Workshop on the Future of Datasets in Vision*, vol. 1, 2015, p. 3.

[62] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[63] H. Zhao *et al.*, "Pyramid Scene Parsing Network", in *CVPR*, 2017.

[64] L.-C. Chen *et al.*, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs", *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[65] M. Larsson, "End-to-end learning of deep structured models for semantic segmentation", PhD thesis, Department of Signals and Systems, Chalmers University of Technology, 2018.

[66] A. Arnab *et al.*, "Conditional random fields meet deep neural networks for semantic segmentation: Combining probabilistic graphical models with deep learning for structured prediction", *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 37–52, 2018.

[67] R. Arandjelović and A. Zisserman, "Visual Vocabulary with a Semantic Twist", in *ACCV*, 2014.

[68] L. Ladickỳ *et al.*, "Joint optimization for object class segmentation and dense stereo reconstruction", *International Journal of Computer Vision*, vol. 100, no. 2, pp. 122–133, 2012.

[69] L. Schneider *et al.*, "Semantic stixels: Depth is not enough", in *Intelligent Vehicles Symposium (IV), 2016 IEEE*, IEEE, 2016, pp. 110–117.

[70] C. Hane *et al.*, "Joint 3d scene reconstruction and class segmentation", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 97–104.

[71] M. Schreiber, C. Knöppel, and U. Franke, "LaneLoc: Lane marking based localization using highly accurate maps", in *IV*, 2013.

[72] R. Spangenberg, D. Goehring, and R. Rojas, "Pole-based localization for autonomous vehicles in urban scenarios", in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, IEEE, 2016, pp. 2161–2166.

[73] T. Naseer *et al.*, "Semantics-aware visual localization under challenging perceptual conditions", in *ICRA*, 2017.

[74] T. Sattler *et al.*, "Image retrieval for image-based localization revisited", in *British Machine Vision Conference*, 2012.

[75] W. Maddern *et al.*, "1 Year, 1000km: The Oxford RobotCar Dataset", *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017. eprint: `http://ijr.sagepub.com/content/early/2016/11/28/0278364916679498.full.pdf+html`.

[76] J. L. Schönberger *et al.*, "Semantic Visual Localization", in *CVPR*, 2018.

Part II

Included Publications

Paper I

Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions

C. Toft, T. Sattler, W. Maddern, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, T. Pajdla and F. Kahl

To be submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI).

Comment: The layout of this paper has been reformatted in order to comply with the rest of the thesis.

Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions

C. Toft, T. Sattler, W. Maddern, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, T. Pajdla and F. Kahl

Abstract

Visual localization enables autonomous vehicles to navigate in their surroundings and augmented reality applications to link virtual to real worlds. Practical visual localization approaches need to be robust to a wide variety of viewing condition, including day-night changes, as well as weather and seasonal variations, while providing highly accurate 6 degree-of-freedom (6DOF) camera pose estimates. In this paper, we extend three publicly available datasets containing images captured under a wide variety of viewing conditions, but lacking camera pose information, with ground truth pose information, making evaluation of the impact of various factors on 6DOF camera pose estimation accuracy possible. We also perform extensive experiments where we evaluate state-of-the-art localization approaches on these datasets. Additionally, we release around half of the poses for all conditions, and keep the remaining half private as a test set, in the hopes that this will stimulate research on long-term visual localization, learned local image features, and related areas. Our datasets are available at `visuallocalization.net`, where we are also hosting a benchmarking server for automatic evaluation of results on the test set.

Estimating the 6DOF camera pose of an image with respect to a 3D scene model is key for visual navigation of autonomous vehicles and augmented/mixed reality devices. Solutions to this *visual localization* problem can also be used to "close loops" in the context of SLAM or to register images to Structure-from-Motion (SfM) reconstructions.

Work on 3D structure-based visual localization has focused on increasing efficiency [1–5], improving scalability and robustness to ambiguous structures [6–9], reducing memory requirements [1, 8, 10], and more flexible scene representations [11]. All these methods utilize local features to establish 2D-3D matches. These correspondences are in turn used to estimate the camera pose. This data association stage is critical as pose estimation fails

Figure 1: Visual localization in changing urban conditions. We present three new datasets, *Aachen Day-Night*, *RobotCar Seasons* (shown) and the *Extended CMU Seasons* for evaluating 6DOF localization against a prior 3D map (top) using registered query images taken from a wide variety of conditions (bottom), including day-night variation, weather, and seasonal changes over long periods of time.

without sufficiently many correct matches. There is a well-known trade-off between discriminative power and invariance for local descriptors. Thus, existing localization approaches will only find enough matches if both the query images and the images used to construct the 3D scene model are taken under similar viewing conditions.

Capturing a scene under all viewing conditions is prohibitive. Thus, the assumption that all relevant conditions are covered is too restrictive in practice. It is more realistic to expect that images of a scene are taken under a single or a few conditions. To be practically relevant, *e.g.*, for life-long localization for self-driving cars, visual localization algorithms need to be robust under varying conditions (*c.f.*Fig. 1). Yet, no work in the literature actually measures the impact of varying conditions on 6DOF pose accuracy.

One reason for this lack of work on visual localization under varying conditions was a lack of suitable benchmark datasets. The standard ap-

proach for obtaining ground truth 6DOF poses for query images is to use SfM. An SfM model containing both the database and query images is built and the resulting poses of the query images are used as ground truth [1, 11, 12]. Yet, this approach again relies on local feature matches and can only succeed if the query and database images are sufficiently similar [13]. The benchmark datasets constructed this way thus tend to only include images that are relatively easy to localize in the first place.

In this paper, we present heavily extended versions of the datasets presented in [14], for benchmarking visual localization under changing conditions. To create these datasets, we heavily rely on human work: We manually annotate matches between images captured under different conditions and verify the resulting ground truth poses. The three complimentary benchmark datasets are based on existing data [15–17]. All consist of a 3D model constructed under one condition and offer query images taken under different conditions: The *Aachen Day-Night* dataset focuses on localizing high-quality night-time images against a day-time 3D model. The *RobotCar Seasons* and the *Extended CMU Seasons* dataset both consider automotive scenarios and depict the same scene under varying seasonal and weather conditions. One challenge of the RobotCar Seasons dataset is to localize low-quality night-time images. The CMU Seasons dataset focuses on the impact of seasons on vegetation and thus the impact of scene geometry changes on localization.

In this paper we make the following **contributions**: *(i)* We create a new outdoor benchmark complete with ground truth and metrics for evaluating 6DOF visual localization under changing conditions such as illumination (day/night), weather (sunny/rain/snow), and seasons (summer/winter). Our benchmark covers multiple scenarios, such as pedestrian and vehicle localization, and localization from single and multiple images as well as sequences. *(ii)* We provide an extensive experimental evaluation of state-of-the-art algorithms from both the computer vision and robotics communities on our datasets. We showed that existing algorithms, including SfM, have severe problems dealing with both day-night changes and seasonal changes in vegetated environments. *(iii)* We showe the value of querying with multiple images, rather than with individual photos, especially under challenging conditions. *(iv)* We have made our benchmarks publicly available at `visuallocalization.net`, where we are also now hosting a benchmarking server for evaluation of localization results on a hidden test set for each dataset. We hope this will stimulate research on long-term visual localization, local image feature learning, and related topics.

A preliminary version of this paper appeared at [14]. Since then, we have extended the datasets and are releasing around half of the ground truth

Dataset	Setting	Image Capture	3D SfM Model (# Sub-Models)	# Images		Condition Changes			6DOF query poses
				Database	Query	Weather	Seasons	Day-Night	
Alderley Day/Night [18]	Suburban	Trajectory		14,607	16,960	✓		✓	
Nordland [19]	Outdoors	Trajectory		143k			✓		
Pittsburgh [20]	Urban	Trajectory		254k	24k				
SPED [21]	Outdoors	Static Webcams		1.27M	120k	✓	✓	✓	
Tokyo 24/7 [22]	Urban	Free Viewpoint		75,984	315			✓	
7 Scenes [23]	Indoor	Free Viewpoint		26,000	17,000				✓
Aachen [15]	Historic City	Free Viewpoint	1.54M / 7.28M (1)	3,047	369				
Cambridge [24]	Historic City	Free Viewpoint	1.89M / 17.68M (5)	6,848	4,081				✓(SfM)
Dubrovnik [1]	Historic City	Free Viewpoint	1.89M / 9.61M (1)	6,044	800				✓(SfM)
Landmarks [9]	Landmarks	Free Viewpoint	38.19M / 177.82M (1k)	204,626	10,000				
Mall [12]	Indoor	Free Viewpoint		682	2296				✓
NCLT [25]	Outdoors & Indoors	Trajectory		about 3.8M			✓		✓
Rome [1]	Landmarks	Free Viewpoint	4.07M / 21.52M (69)	15,179	1000				
San Francisco [9, 11, 26]	Urban	Free Viewpoint	30M / 149M (1)	610,773	442				✓(SfM)
Vienna [27]	Landmarks	Free Viewpoint	1.12M / 4.85M (3)	1,324	266				
Aachen Day-Night [14]	Historic City	Free Viewpoint	1.65M / 10.55M (1)	4,328	922			✓	✓
RobotCar Seasons (updated)	Urban	Trajectory	6.77M / 36.15M (49)	27,180	5,616	✓	✓	✓	✓
Extended CMU Seasons (new)	Suburban	Trajectory	3.37M / 17.17M (24)	60,937	56,613	✓	✓		✓

Table 1: Comparison with existing benchmarks for place recognition and visual localization. "Condition Changes" indicates that the viewing conditions of the query images and database images differ. For some datasets, images were captured from similar camera trajectories. If SfM 3D models are available, we report the number of sparse 3D points and the number of associated features. Only our datasets provide a diverse set of changing conditions, reference 3D models, and most importantly ground truth 6DOF poses for the query images.

poses for all environmental conditions for the CMU Seasons and RobotCar Seasons datasets. Particularly, the CMU Seasons dataset has been extended with around 40% more images. These new images are primarily from areas with heavy vegetation, and are thus very challenging since this is the type of scenery that all evaluated localization methods struggle with. We hope these new datasets will stimulate research on long-term visual localization, local image feature learning, and related topics.

1 Related Work

Localization benchmarks. Tab. 1 compares our benchmark datasets with existing datasets for both visual localization and place recognition. Datasets for place recognition [18, 19, 21, 22, 28] often provide query images captured under different conditions compared to the database images. However, they neither provide 3D models nor 6DOF ground truth poses. Thus, they cannot be used to analyze the impact of changing conditions on pose estimation accuracy. In contrast, datasets for visual localization [1, 9, 11, 12, 15, 23, 24, 26, 27] often provide ground truth poses. However, they do not exhibit strong changes between query and database images due to relying on feature matching for ground truth generation. A notable exception is the Michigan North Campus Long-Term (NCLT) dataset [25], providing images captured over long period of time and ground truth obtained via GPS and LIDAR-based SLAM. Yet, it does not cover all viewing condi-

tions captured in our datasets, *e.g.*, it does not contain any images taken at night or during rain. To the best of our knowledge, ours are the first datasets providing both a wide range of changing conditions and accurate 6DOF ground truth. Thus, ours is the first benchmark that measures the impact of changing conditions on pose estimation accuracy.

Datasets such as KITTI [29], TorontoCity [30], or the Málaga Urban dataset [31] also provide street-level image sequences. Yet, they are less suitable for visual localization as only few places are visited multiple times.

3D structure-based localization methods [1, 2, 8, 9, 32–34] establish correspondences between 2D features in a query image and 3D points in an SfM point cloud via descriptor matching. These 2D-3D matches are then used to estimate the query's camera pose. Descriptor matching can be accelerated by prioritization [1, 2, 35] and efficient search algorithms [4, 36]. In large or complex scenes, descriptor matches become ambiguous due to locally similar structures found in different parts of the scene [9]. This results in high outlier ratios of up to 99%, which can be handled by exploiting co-visibility information [8, 9, 34] or via geometric outlier filtering [32, 33, 37].

We evaluate *Active Search* [2] and the *City-Scale Localization* approach [32], a deterministic geometric outlier filter based on a known gravity direction, as representatives for efficient respectively scalable localization methods.

2D image-based localization methods approximate the pose of a query image using the pose of the most similar photo retrieved from an image database. They are often used for place recognition [21, 22, 38–41] and loop-closure detection [42–44]. They remain effective at scale [11, 28, 39, 45] and can be robust to changing conditions [11, 21, 22, 38, 40, 46]. We evaluate two compact VLAD-based [47] image-level representations: DenseVLAD [22] aggregates densely extracted SIFT descriptors [48, 49] while NetVLAD [38] uses learned features. Both are robust against day-night changes [22, 38] and work well at large-scale [11].

We also evaluate the de-facto standard approach for loop-closure detection in robotics [50, 51], where robustness to changing conditions is critical for long-term autonomous navigation [18, 21, 22, 40, 46, 52]: FAB-MAP [42] is an image retrieval approach based on the Bag-of-Words (BoW) paradigm [53] that explicitly models the co-occurrence probability of different visual words.

Sequence-based approaches for image retrieval are used for loop-closure detection in robotics [18, 54, 55]. Requiring a matched sequence of images in the correct order significantly reduces false positive rates compared to single-image retrieval approaches, producing impressive results including direct

day-night matches with SeqSLAM [18]. We evaluate OpenSeqSLAM [19] on our benchmark.

Multiple cameras with known relative poses can be modelled as a generalized camera [56], *i.e.*, a camera with multiple centers of projections. Approaches for absolute pose estimation for both multi-camera systems [57] and camera trajectories [58] from 2D-3D matches exist. Yet, they have never been applied for localization in changing conditions. In this paper, we show that using multiple images can significantly improve performance in challenging scenarios.

Learning-based localization methods have been proposed to solve both loop-closure detection [21, 40, 59, 60] and pose estimation [24, 61–63]. They learn features with stable appearance over time [21, 46, 64], train classifiers for place recognition [52, 65–67], and train CNNs to regress 2D-3D matches [23, 68, 69] or camera poses [24, 61, 62].

2 Benchmark Datasets for 6DOF Localization

This section describes the creation of our three new benchmark datasets. Each dataset is constructed from publicly available data, allowing our benchmarks to cover multiple geographic locations. We add ground truth poses for all query images and build reference 3D models (*c.f.* Fig. 3) from images captured under a single condition.

All three datasets present different challenges. The *Aachen Day-Night* dataset focuses on localizing night-time photos against a 3D model built from day-time imagery. The night-time images, taken with a mobile phone using software HDR post-processing, are of high quality. The dataset represents a scenario where images are taken with hand-held cameras, *e.g.*, an augmented reality application.

Both the *RobotCar Seasons* and the *Extended CMU Seasons* datasets represent automotive scenarios, with images captured from a car. In contrast to the Aachen Day dataset, both datasets exhibit less variability in viewpoints but a larger variance in viewing conditions. The night-time images from the RobotCar dataset were taken from a driving car with a consumer camera with auto-exposure. This results in significantly less well-lit images exhibiting motion blur, *i.e.*, images that are significantly harder to localize (*c.f.*Fig. 2).

The RobotCar dataset depicts a mostly urban scene with rather static scene geometry. In contrast, the CMU dataset contains a significant amount of vegetation. The changing appearance and geometry of the vegetation, due to seasonal changes, is the main challenge of this dataset.

	reference model				query images
	# images	# 3D points	# features	condition	conditions (# images)
Aachen Day-Night	4,328	1.65M	10.55M	day	day (824), night (98)
RobotCar Seasons	20,862	6.77M	36.15M	overcast (November)	dawn (681), dusk (591), night (678), night+rain (609), rain (615), overcast summer / winter (633 / 492), snow (645), sun (672)
Extended CMU Seasons	10,338	3.37M	17.17M	sun / no foliage (April)	sun (16,300), low sun (21,629), overcast (8,179), clouds (10,270), foliage (24,876), mixed foliage (20,848), no foliage (10,654) urban (18,307), suburban (21,512), park (16,559)

Table 2: Detailed statistics for the three benchmark datasets proposed in this paper. For each dataset, a reference 3D model was constructed using images taken under the same reference condition, *e.g.*, "overcast" for the RobotCar Seasons dataset.

2.1 The Aachen Day-Night Dataset

Our Aachen Day-Night dataset is based on the Aachen localization dataset from [15]. The original dataset contains 4,479 reference and 369 query images taken in the old inner city of Aachen, Germany. It provides a 3D SfM model but does not have ground truth poses for the queries. We augmented the original dataset with day- and night-time queries captured using standard consumer phone cameras.

To obtain ground truth poses for the day-time queries, we used COLMAP [70] to create an intermediate 3D model from the reference and day-time query images. The scale of the reconstruction is recovered by aligning it with the geo-registered original Aachen model. As in [1], we obtain the reference model for the Aachen Day-Night dataset by removing the day-time query images. 3D points visible in only a single remaining camera were removed as well [1]. The resulting 3D model has 4,328 reference images and 1.65M 3D points triangulated from 10.55M features.

Ground truth for night-time queries. We captured 98 night-time query images using a Google Nexus5X phone with software HDR enabled. Attempts to include them in the intermediate model resulted in highly inaccurate camera poses due to a lack of sufficient feature matches. To obtain ground truth poses for the night-time queries, we thus hand-labelled 2D-3D matches. We manually selected a day-time query image taken from a similar viewpoint for each night-time query. For each selected day-time query, we projected its visible 3D points from the intermediate model into it. Given these projections as reference, we manually labelled 10 to 30 corresponding pixel positions in the night-time query. Using the resulting 2D-3D matches and the known intrinsics of the camera, we estimate the camera poses using a 3-point solver [71, 72] and non-linear pose refinement.

To estimate the accuracy for these poses, we measure the mean reprojection error of our hand-labelled 2D-3D correspondences (4.33 pixels for 1600x1200 pixel images) and the pose uncertainty. For the latter, we compute multiple poses from a subset of the matches for each image and measure

Figure 2: Example query images for *Aachen Day-Night* (top), *RobotCar Seasons* (middle) and *CMU Seasons* datasets (bottom).

the difference in these poses to our ground truth poses. The mean median position and orientation errors are 36cm and 1°. The absolute pose accuracy that can be achieved by minimizing a reprojection error depends on the distance of the camera to the scene. Given that the images were typically taken 15 or more meters from the scene, we consider the ground truth poses to be reasonably accurate.

2.2 The RobotCar Seasons Dataset

Our RobotCar Seasons dataset is based on a subset of the publicly available Oxford RobotCar Dataset [16]. The original dataset contains over 20M images recorded from an autonomous vehicle platform over 12 months in Oxford, UK. Out of the 100 available traversals of the 10km route, we select one reference traversal in overcast conditions and nine query traversals that cover a wide range of conditions (*c.f.*Tab. 1). All selected images were taken with the three synchronized global shutter Point Grey Grasshopper2

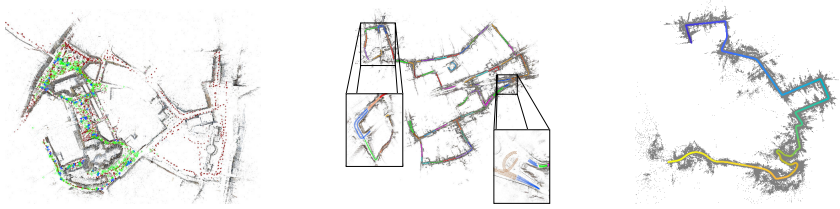


Figure 3: 3D models of the *Aachen Day-Night* (left, showing database (red), day-time query (green), and night-time query images (blue)), *RobotCar Seasons* (middle), and *Extended CMU Seasons* (right) datasets. For RobotCar and CMU, the colors encode the individual submaps.

cameras mounted to the left, rear, and right of the car. Both the intrinsics of the cameras and their relative poses are known.

The reference traversal contains 26,121 images taken at 8,707 positions, with 1m between successive positions. Building a single consistent 3D model from this data is very challenging, both due to sheer size and the lack of visual overlap between the three cameras. We thus built 49 non-overlapping local submaps, each covering a 100m trajectory. For each submap, we initialized the database camera poses using vehicle positions reported by the inertial navigation system (INS) mounted on the RobotCar. We then iteratively triangulated 3D points, merged tracks, and refined both structure and poses using bundle adjustment. The scale of the reconstructions was recovered by registering them against the INS poses. The reference model contains all submaps and consists of 20,862 reference images and 6.77M 3D points triangulated from 36.15M features.

We obtained images from the other traversals by selecting reference positions inside the 49 submaps and gathering all images from the nine other traversals with INS poses within 10m of one of these positions. This resulted in 11,934 images in total, where triplets of images were captured at 3,978 distinct locations. These images were grouped into 460 temporal sequences based on the timestamps of the images.

We have also, in addition to the poses of the images from the reference traversal, also publicly released around half of the camera poses for the images from the other traversals (and thus captured during different conditions), for a total of 27,180 images. The camera poses of the remaining 5,616 images are used as a hidden test set in the long-term visual localization benchmark for the RobotCar Seasons dataset.

Ground truth poses for the queries. Due to GPS drift, the INS poses for these other traversals cannot be directly used as ground truth. Again, there are not enough feature matches between day- and night-time images for SfM. We thus used the LIDAR scanners mounted to the vehicle to build

local 3D point clouds for each of the 49 submaps under each condition. These models were then aligned to the LIDAR point clouds of the reference trajectory using ICP [73]. Many alignments needed to be manually adjusted to account for changes in scene structure over time (often due to building construction and road layout changes). The final median RMS errors between aligned point clouds was under 0.10m in translation and 0.5° in rotation across all locations. The alignments provided ground truth poses for these images.

2.3 The CMU Seasons Dataset

The Extended CMU Seasons Dataset is based on a subset of the CMU Visual Localization Dataset [17], which contains more than 100K images recorded by the Computer Vision Group at Carnegie Mellon University over a period of 12 months in Pittsburgh, PA, USA. The images were collected using a rig of two cameras mounted at approximately 45 degrees forward/left and forward/right angles on the roof of an SUV. The vehicle traversed an 8.5 km long route through central and suburban Pittsburgh 16 times with a spacing in time of between 2 weeks up to 2 months. Out of the 16 traversals, we selected the one from April 4 as the reference, and then 11 query traversals were selected such that they cover the range of variations in seasons and weather that the data set contains.

As with the RobotCar Seasons dataset, we publicly release all images and corresponding ground truth poses for reference traversal, in addition to around half of the ground truth poses from the other traversals, for a total of 60,937 images. The remaining half remain private as a test set for benchmarking purposes.

Ground truth poses for the queries. As with the RobotCar dataset, the GPS is not accurate enough and the CMU dataset is also too large to build one 3D model from all the images. The full sequences were split up into 24 shorter sequences, each containing about 250 consecutive vehicle poses. For each short sequence, a 3D model was built using bundle adjustment of SIFT points tracked over several image frames. The resulting submaps of the reference route were merged with the corresponding submaps from the other traversals by using global bundle adjustment and manually annotating image correspondences across sequences collected during different dates. Reprojection errors are within a few pixels for all 3D points and the distances between estimated camera positions and expected ones (based on neighbouring cameras) are under 0.10m. The resulting reference model consists of 3.37M 3D points triangulated from 17.17M features in 10,338 database images.

3 Benchmark Setup

We evaluate state-of-the-art localization approaches on our new benchmark datasets to measure the impact of changing conditions on camera pose estimation accuracy and to understand how hard robust long-term localization is.

Evaluation measures. We measure the *pose accuracy* of a method by the deviation between the estimated and the ground truth pose. The *position error* is measured as the Euclidean distance $\|c_{\text{est}} - c_{\text{gt}}\|_2$ between the estimated c_{est} and the ground truth position c_{gt}. The absolute *orientation error* $|\alpha|$, measured as an angle in degrees, is computed from the estimated and ground truth camera rotation matrices $\mathtt{R}_{\text{est}}$ and $\mathtt{R}_{\text{gt}}$. We follow standard practice [74] and compute $|\alpha|$ as $2\cos(|\alpha|) = \text{trace}(\mathtt{R}_{\text{gt}}^{-1}\mathtt{R}_{\text{est}}) - 1$, *i.e.*, we measure the minimum rotation angle required to align both rotations [74].

We measure the percentage of query images localized within Xm and Y° of their ground truth pose. We define three pose accuracy intervals by varying the thresholds: *High-precision* (0.25m, 2°), *medium-precision* (0.5m, 5°), and *coarse-precision* (5m, 10°). These thresholds were chosen to reflect the high accuracy required for autonomous driving. We use the intervals (0.5m, 2°), (1m, 5°), (5m, 10°) for the Aachen night-time queries to account for the higher uncertainty in our ground truth poses. Still, all regimes are more accurate than consumer-grade GPS systems.

Evaluated algorithms. As discussed in Sec. 2, we evaluate a set of state-of-the-art algorithms covering the most common types of localization approaches: From the class of 3D structure-based methods, we use *Active Search* (AS) [11] and *City-Scale Localization* (CSL) [32]. From the class of 2D image retrieval-based approaches, we use *DenseVLAD* [22], *NetVLAD* [38], and *FAB-MAP* [42].

In order to measure the benefit of using multiple images for pose estimation, we evaluate two approaches: *OpenSeqSLAM* [19] is based on image retrieval and enforces that the images in the sequence are matched in correct order. Knowing the relative poses between the query images, we can model them as a generalized camera [56]. Given 2D-3D matches per individual image (estimated via Active Search), we estimate the pose via a generalized absolute camera pose solver [57] inside a RANSAC loop. We denote this approach as *Active Search+GC* (AS+GC). We mostly use ground truth query poses to compute the relative poses that define the generalized cameras[1]. Thus, AS+GC provides an upper bound on the number of images that can

[1]Note that Active Search+GC only uses the relative poses between the query images to define the geometry of a generalized camera. It does *not* use any information about the absolute poses of the query images.

be localized when querying with generalized cameras.

The methods discussed above all perform localization from scratch without any prior knowledge about the pose of the query. In order to measure how hard our datasets are, we also implemented two *optimistic baselines*. Both assume that a set of relevant database images is known for each query. Both perform pairwise image matching and use the known ground truth poses for the reference images to triangulate the scene structure. The feature matches between the query and reference images and the known intrinsic calibration are then be used to estimate the query pose. The first optimistic baseline, *LocalSfM*, uses upright RootSIFT features [48, 49]. The second uses upright CNN features densely extracted on a regular grid. We use the same VGG-16 network [75] as NetVLAD. The *DenseSfM* method uses coarse-to-fine matching with conv4 and conv3 features.

We select the relevant reference images for the two baselines as follows: For Aachen, we use the manually selected day-time image (*c.f.*Sec. 2.1) to select up to 20 reference images sharing the most 3D points with the selected day-time photo. For RobotCar and CMU, we use all reference images within 5m and 135° of the ground truth query pose.

We evaluated *PoseNet* [24] but were not able to obtain competitive results. We also attempted to train *DSAC* [69] on KITTI but were not able to train it. Both PoseNet and DSAC were thus excluded from further evaluations.

4 Experimental Evaluation

This section presents the second main contribution of this paper, a detailed experimental evaluation on the effect of changing conditions on the pose estimation accuracy of visual localization techniques. In the following, we focus on pose accuracy. For more details on the algorithms and their specific settings, see Sec. 5. In Fig. 5, cumulative distribution functions of the translational error of the methods are shown for different conditions in the three datasets.

	Aachen		**Extended CMU**					
	day	night	foliage	mixed foliage	no foliage	urban	suburban	park
m	.25/.50/5.0	0.5/1.0/5.0	.25/.50/5.0	.25/.50/5.0	.25/.50/5.0	.25/.50/5.0	.25/.50/5.0	.25/.50/5.0
deg	2/5/10	2/5/10	2/5/10	2/5/10	2/5/10	2/5/10	2/5/10	2/5/10
Active Search	57.3 /83.7 /96.6	19.4 /30.6 /43.9	/ /	/ /	/ /	/ /	/ /	/ /
CSL	52.3 /80.0 /94.3	24.5 /33.7 /49.0	47.0 / 50.2 / 55.3	52.4 / 56.1 / 62.0	80.3 / 83.2 / 86.5	71.2 / 74.6 / 78.7	57.8 / 61.6 / 67.5	34.4 / 37.0 / 42.2
DenseVLAD	0.0 /0.1 /22.8	0.0 /2.0 /14.3	7.4 /21.1 /68.0	8.5 /24.5 /73.0	10.0 / 32.7 /88.0	14.7 /36.3 /83.9	5.3 /18.7 /73.9	5.2 /19.2 / 62.0
NetVLAD	0.0 /0.2 /18.9	0.0 /2.0 /12.2	6.2 / 18.5 / 74.3	5.8 / 17.6 / 71.1	6.7 / 20.9 / 79.4	12.2 / 31.5 / 89.8	3.7 / 13.9 / 74.7	2.6 / 10.4 / 55.9
FABMAP	0.0 /0.0 /4.6	0.0 /0.0 /0.0	/ /	/ /	/ /	/ /	/ /	/ /
LocalSfM		36.7 /54.1 /72.4	/ /	/ /	/ /	/ /	/ /	/ /
DenseSfM		39.8 /60.2 /84.7						
AS+GC(seq)			/ /	/ /	/ /	/ /	/ /	/ /

Table 3: Evaluation on the Aachen Day-Night dataset and a subset of the conditions of the Extended CMU Seasons dataset.

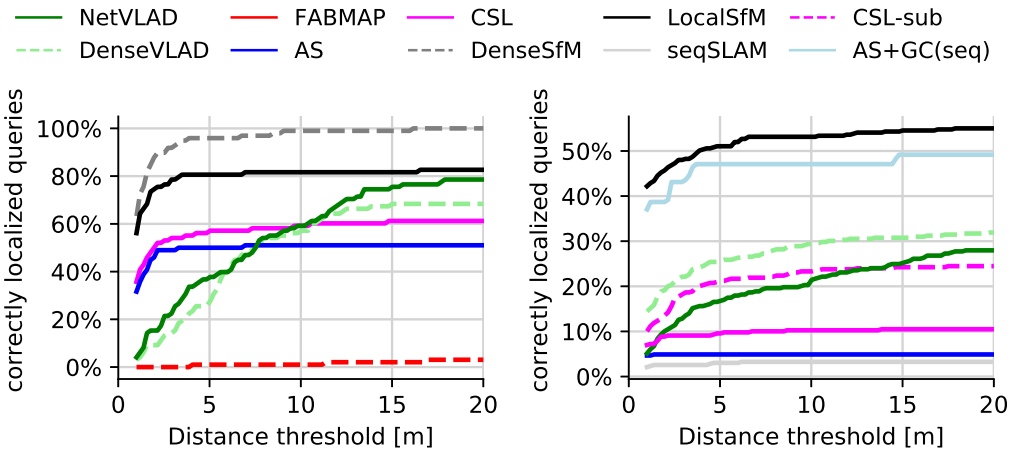


Figure 4: Cumulative distribution of position errors for the night-time queries of the Aachen (left) and RobotCar (right) datasets.

4.1 Evaluation on the Aachen Day-Night Dataset

The focus of the Aachen Day-Night dataset is on benchmarking the pose accuracy obtained by state-of-the-art methods when localizing night-time queries against a 3D model constructed from day-time imagery. In order to put the results obtained for the night-time queries into context, we first evaluate a subset of the methods on the 824 day-time queries. As shown in Tab. 3, the two structure-based methods are able to estimate accurate camera poses and localize nearly all images within the coarse-precision regime. We conclude that the Aachen dataset is not particularly challenging for the day-time query images.

Night-time queries. Tab. 3 also reports the results obtained for the night-time queries. We observe a significant drop in pose accuracy for both Active Search and CSL, down from above 50% in the high-precision regime to less than 50% in the coarse-precision regime. Given that the night-time queries were taken from similar viewpoints as the day-time queries, this drop is solely caused by the day-night change.

CSL localizes more images compared to Active Search (AS). This is not surprising since CSL also uses matches that were rejected by AS as too ambiguous. Still, there is a significant difference to LocalSfM. CSL and AS both match features against the full 3D model while LocalSfM only considers a small part of the model for each query. This shows that global matching sufficiently often fails to find the correct nearest neighbors, likely caused by significant differences between day-time and night-time descriptors.

Fig. 4(left) shows the cumulative distribution of position errors for the night-time queries and provides interesting insights: LocalSfM, despite know-

	day conditions							night conditions	
	dawn	dusk	OC-summer	OC-winter	rain	snow	sun	night	night-rain
m	.25 / .50 / 5.0	.25 / .50 / 5.0	.25 / .50 / 5.0	.25 / .50 / 5.0	.25 / .50 / 5.0	.25 / .50 / 5.0	.25 / .50 / 5.0	.25 / .50 / 5.0	.25 / .50 / 5.0
deg	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10
ActiveSearch	42.7 / 87.7 / 98.7	57.9 / 86.8 / 100	28.9 / 73.9 / 94.3	39 / 90.2 / 100	66.3 / 91.7 / 99.5	42.3 / 84.7 / 98.6	33 / 53.1 / 71.9	0.9 / 2.2 / 3.5	1 / 3.9 / 6.4
CSL	54.2 / 89.4 / 96.9	75.1 / 95.4 / 100	37.4 / 82.9 / 91.5	48.2 / 96.3 / 100	73.2 / 94.6 / 100	61.4 / 94.9 / 97.2	33.9 / 52.7 / 71	0.4 / 1.3 / 6.2	1 / 5.4 / 12.8
DenseVLAD	15.4 / 45.8 / 97.4	7.6 / 35.5 / 98.5	9 / 30.3 / 88.2	2.4 / 28.7 / 97.6	13.7 / 50.7 / 100	10.2 / 38.1 / 93.5	8 / 22.3 / 78.1	0.9 / 4.4 / 24.3	2.5 / 5.9 / 25.1
NetVLAD	10.1 / 28.2 / 87.7	4.6 / 25.4 / 97.5	10 / 35.1 / 97.6	2.4 / 28.7 / 100	12.2 / 46.8 / 100	8.8 / 32.6 / 95.3	8.5 / 22.8 / 88.8	0 / 0.9 / 18.1	0.5 / 2 / 13.3
FABMAP	2.2 / 5.3 / 10.6	2.5 / 18.3 / 57.4	0.9 / 9.5 / 30.8	0.6 / 14 / 46.3	12.2 / 40.5 / 92.2	3.3 / 8.8 / 28.4	0 / 0 / 0.9	0 / 0 / 0	0 / 0 / 0

Table 4: Evaluation on the RobotCar Seasons dataset. We report the percentage of queries localized within the three thresholds.

ing relevant reference images for each query, completely fails to localize about 20% of all queries. This is caused by a lack of correct feature matches for these queries, either due to failures of the feature detector or descriptor. DenseSfM skips feature detection and directly matches densely extracted CNN descriptors (which encode higher-level information compared to the gradient histograms used by RootSIFT). This enables DenseSfM to localize more images at a higher accuracy, resulting in the best performance on this dataset. Still, there is significant room for improvement, even in the coarse-precision regime (*c.f.*Tab. 3). Also, extracting and matching dense descriptors is a time-consuming task.

4.2 Evaluation on the RobotCar Seasons Dataset

The focus of the RobotCar Seasons dataset is to measure the impact of different seasons and illumination conditions on pose estimation accuracy in an urban environment.

Tab. 4 shows that changing day-time conditions have only a small impact on pose estimation accuracy for all methods. The reason is that seasonal changes have little impact on the building facades that are dominant in most query images. The exception is the "sun" condition, where we observed overexposed images caused by direct sunlight (*c.f.*Fig. 1). Thus, fewer features can be found for Active Search and CSL and the global image descriptors used by the image retrieval approaches are affected as well.

On the Aachen Day-Night dataset, we observed that image retrieval-based methods (DenseVLAD and NetVLAD) consistently performed worse than structure-based methods (Active Search, CSL, LocalSfM, and DenseSfM). For the RobotCar dataset, NetVLAD and DenseVLAD essentially achieve the same coarse-precision performance as Active Search and CSL. This is caused by the lower variation in viewpoints as the car follows the same road.

Compared to Aachen, there is an even stronger drop in pose accuracy between day and night for the RobotCar dataset. All methods fail to localize a significant number of queries for both the high- and medium-precision regimes. Interestingly, DenseVLAD and NetVLAD outperform all other

	all day	all night
m	.25 / .50 / 5.0	.25 / .50 / 5.0
deg	2 / 5 / 10	2 / 5 / 10
ActiveSearch	44.1 / 80.6 / 94.3	0.9 / 3 / 4.9
CSL	54.5 / 85.9 / 93.3	0.7 / 3.3 / 9.3
ActiveSearch+GC (triplet)	54.4 / 90.9 / 98.7	1.6 / 7.9 / 12.1
ActiveSearch+GC (sequence, GT)	53.2 / 93.3 / 100	5.4 / 25.6 / 47.1
seqSLAM	1 / 6 / 15.9	0.5 / 1.4 / 3

Table 5: Using multiple images for pose estimation (ActiveSeach+GC) on the RobotCar Seasons dataset.

methods in the coarse-precision regime (*c.f.* Fig. 4(right)). This shows that their global descriptors still encode distinctive information even if local feature matching fails. The better performance of all methods under "night+rain" compared to "night" comes from the autoexposure of the RobotCar's cameras. A longer exposure is used for the "night", leading to significant motion blur.

Additionally, compared to the original test set of the RobotCar Seasons from [14], the new test set seems to be somewhat easier: the performance of all methods increase by a several percent compared to the results in [14].

Multi-image queries. The RobotCar is equipped with three synchronized cameras and captures sequences of images for each camera. Rather than querying with only a single image, we can thus also query with multiple photos. Tab. 5 shows the results obtained with seqSLAM (which uses temporal sequences of all images captured by the three cameras) and Active Search+GC. For the latter, we query with triplets of images taken at the same time as well as with temporal sequences of triplets. For the triplets, we use the known extrinsic calibration between the three cameras mounted on the car. For the temporal sequences, we use relative poses obtained from the ground truth (GT) absolute poses. For readability, we only show the results summarized for day- and night-conditions.

Tab. 5 shows that Active Search+GC consistently outperforms single image methods in terms of pose accuracy. Active Search+GC is able to accumulate correct matches over multiple images. This enables Active Search+GC to succeed even if only a few matches are found for each individual image. Naturally, the largest gain can be observed when using multiple images in a sequence.

Location priors. In all previous experiments, we considered the full RobotCar 3D model for localization. However, it is not uncommon in outdoor settings to have a rough prior on the location at which the query image was taken. We simulate such a prior by only considering the submodel relevant to a query rather than the full model. While we observe only

		RobotCar - all night
	m	.25 / .50 / 5.0
	deg	2 / 5 / 10
2*ActiveSearch	full model	0.9 / 3 / 4.9
	sub-model	4.4 / 11.7 / 16.6
2*CSL	full model	0.7 / 3.3 / 9.3
	sub-model	0.5 / 4.7 / 18.4
2*ActiveSearch+GC (triplet)	full model	1.6 / 7.9 / 12.1
	sub-model	9.3 / 21.2 / 29.4
2*ActiveSearch+GC (sequence, GT)	full model	5.4 / 25.6 / 47.1
	sub-model	17.7 / 42.7 / 64.1
2*ActiveSearch+GC (sequence, VO)	full model	1.4 / 11.2 / 24.2
	sub-model	3.7 / 16.1 / 48
LocalSfM	sub-model	20 / 35.9 / 49.9

Table 6: Using location priors to query only submodels rather than the full RobotCar Seasons dataset for night-time queries.

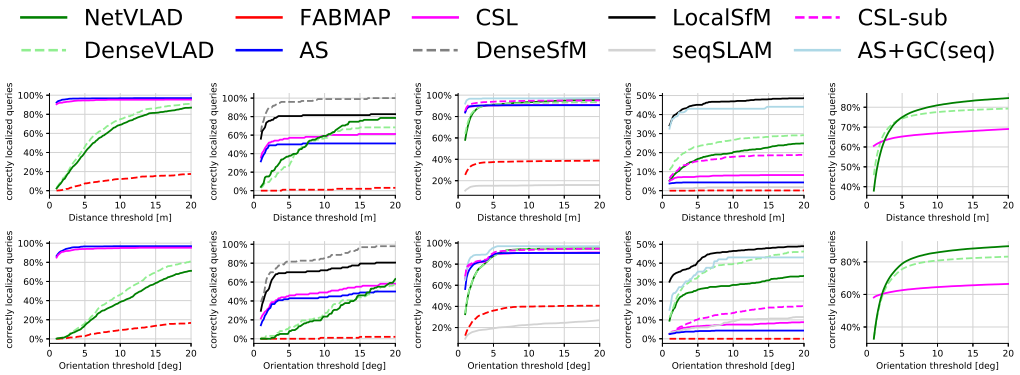
a small improvement for day-time queries, localizing night-time queries significantly benefits from solving an easier matching problem (*c.f.*Tab. 6). For completeness, we also report results for LocalSfM, which also considers only a small part of the model relevant to a query. Active Search+GC outperforms LocalSfM on this easier matching task when querying with sequences in the coarse regime. This is due to not relying on one single image to provide enough matches.

One drawback of sequence-based localization is that the relative poses between the images in a sequence need to be known quite accurately. Tab. 6 also reports results obtained when using our own multi-camera visual odometry (VO) system to compute the relative poses. While performing worse compared to ground truth relative poses, this more realistic baseline still outperforms methods using individual images. The reasons for the performance drop are drift and collapsing trajectories due to degenerate configurations.

4.3 Evaluation on the Extended CMU Seasons Dataset

Compared to the urban scenes shown in the other datasets, significant parts of the Extended CMU Seasons dataset show suburban or park regions. Seasonal changes can drastically affect the appearance of such regions. In the following, we thus focus on these conditions. For each query image, we only consider its relevant sub-model.

Tab. 3 evaluates the impact of changes in foliage and of different regions on pose accuracy. The reference condition for the Extended CMU Seasons dataset does not contain foliage. Thus, other conditions for which foliage is also absent lead to the most accurate poses. Interestingly, DenseVLAD and NetVLAD achieve a better performance than Active Search and CSL for



(a) Aachen - day (b) Aachen - night (c) RobotCar - day (d) RobotCar - night (e) CMU - all

Figure 5: Cumulative distribution of position and orientation errors for the three datasets.

the medium- and coarse-precision regimes under the "Foliage" and "Mixed Foliage" conditions. This again shows that global image-level descriptors can capture information lost by local features.

We observe a significant drop in pose accuracy in both suburban and park regions. This is caused by the dominant presence of vegetation, leading to many locally similar (and thus globally confusing) features. Again, we notice that DenseVLAD and NetVLAD are able to coarsely localize more queries compared to the feature-based methods.

5 Details on the Evaluated Algorithms

This section provides a detailed description, including parameter settings, of the state-.

5.1 3D Structure-based Localization

Active Search (AS). Active Search [2] accelerates 2D-3D descriptor matching via a prioritization scheme. It uses a visual vocabulary to quantize the descriptor space. For each query feature, it determines how many 3D point descriptors are assigned to the feature's closest visual word. This determines the number of descriptor comparisons needed for matching this feature. Active Search then matches the features in ascending order of the number of required descriptor comparisons. If a 2D-to-3D match is found, Active Search attempts to find additional 3D-to-2D correspondences for the 3D points surrounding the matching point. Correspondence search terminates once 100 matches have been found.

For the Aachen Day-Night dataset, we trained a visual vocabulary containing 100k words using approximate k-means clustering [76] on all upright RootSIFT [48, 49] descriptors found in 1,000 randomly selected database images contained in the 3D model. Similarly, we trained a vocabulary containing 10k words for the RobotCar Seasons dataset from the descriptors found in 1,000 randomly selected images contained in the reference 3D model.

We use calibrated cameras rather than simultaneously estimating each camera's extrinsic and intrinsic parameters. We thereby exploit the known intrinsic calibrations provided by the intermediate model of the Aachen Day-Night dataset[2] and the known intrinsics of the RobotCar Seasons.

Besides training new vocabularies and using calibrated cameras, we only changed the threshold on the re-projection error used by RANSAC to distinguish between inliers and outliers. For the Aachen Day-Night dataset, we used a threshold of 10 pixels while we used 5 pixels for both the RobotCat Seasons.Otherwise, we used the standard parameters of Active Search.

City-Scale Localization (CSL). The City-Scale Localization algorithm [32] is an outlier rejection algorithm, *i.e.*, it is a robust localization algorithm that can prune guaranteed outlier correspondences from a given set of 2D-3D correspondences. CSL is based on the following central insight: If the gravity direction and an approximate height of the camera above the ground plane are known, it is possible to calculate an upper bound for the maximum number of inliers that any solution containing a given 2D-3D correspondence as an inlier can have. At the same time, CSL also computes a lower bound on the number of inliers for a given correspondence by computing a pose for which this correspondence is an inlier. CSL thus computes this upper bound for each 2D-3D match and, similar to RANSAC, continuously updates the best pose found so far (which provides a lower bound on the number of inliers that can be found). All correspondences with an upper bound on the maximum number of inliers that is below the number of inliers in the current best solution can be permanently discarded from further consideration. When outliers have been discarded, three-point RANSAC [71, 72] is performed on the remaining correspondences. Notice that, unlike RANSAC, the outlier filter used by CSL is deterministic. The computational complexity of the filter is $\mathcal{O}(N^2 \log N)$, where N is the number of available 2D-3D correspondences.

In order to obtain an estimate for the gravity direction, we follow [32] and add noise to the gravity direction obtained from the ground truth poses.

[2]Some of the day-time queries were taken with the same camera as the night-time queries and we enforced that the images taken with the same camera have consistent intrinsics. Thus, the intermediate model provides the intrinsic calibration of the night-time queries.

Parameter	Value
Feature Type	Dense RootSIFT
Vocabulary Size (trained on SF)	128
Descriptor Dimension (after PCA & whitening)	4,096

Table 7: DenseVLAD parameters.

Parameter	Value
Network model (trained on Pitts30k)	VGG-16 + NetVLAD + whitening
Descriptor Dimension	4,096

Table 8: NetVLAD parameters.

CSL iterates over a range of plausible height values, similar to [33]. In these experiments, the height values cover an interval five meters high. This interval is centered on the camera height of the ground truth pose, with added noise. In the Aachen experiments, the height interval is divided into nine sections, and for the Oxford and CMU experiments, the height interval is divided into three sections.

The 2D-3D correspondences are generated by matching the descriptors of all detected features in the query image to the descriptors of the 3D points using approximate nearest neighbour search. To account for the fact that each 3D point is associated with multiple descriptor, the 3D points are each assigned a single descriptor vector equal to the mean of all its corresponding descriptors. This matching strategy yields the same number of correspondences as the number of detected features.

As with Active Search, we use a re-projection error threshold of 10 pixels for the Aachen Day-Night dataset and 5 pixels for both the RobotCat Seasons and the Extended CMU Seasons datasets.

5.2 2D Image-based Localization

DenseVLAD and NetVLAD. We use the original implementations of DenseVLAD [77] and NetVLAD [38] provided by the authors. Images were processed at their original resolution unless any dimension exceeded 1920 pixels. For DenseVLAD, we used the Dense SIFT implementation, followed by RootSIFT normalization [48], available in VLFeat [78]. The visual vocabulary used consisted of 128 visual words (centroids) pre-computed on the San-Francisco (SF) dataset [79], *i.e.*, we used a general vocabulary trained

Parameter	Value
Feature Type	UprightSURF128
Aachen Vocabulary Size	3585
RobotCar Vocabulary Size	5031
$p\left(z_i \mid \bar{e}_i\right)$	0
$p\left(\bar{z}_i \mid e_i\right)$	0.61
$p\left(L_{\text{new}} \mid Z^{k-1}\right)$	0.9

Table 9: FAB-MAP parameters.

on a different yet similiar dataset. For NetVLAD we used the pre-computed network "Pitts30k" trained on the Pittsburgh time-machine street-view image dataset [38]. The network is therefore not fine-tuned on our datasets, *i.e.*, we again used a general network trained on a different city.

Given a DenseVLAD or NetVLAD descriptor, we find the most similar reference image by exhaustive nearest neighbor search. While this stage could be accelerated by approximate search, we found this to be unnecessary as the search for a single query descriptor typically takes less than 20ms.

Tables 7 and 8 summarize the parameters used for DenseVLAD and NetVLAD in our experiments.

FAB-MAP. For FAB-MAP [42], we trained a separate vocabulary for each location using Modified Sequential Clustering [80] on evenly spaced database images, resulting in 3,585 visual words for Aachen Day-Night, 5,031 for RobotCar Seasons. A Chow-Liu tree was built for each dataset using the Bag-of-Words generated for each database image using the vocabulary. We used the mean field approximation for the new place likelihood (as additional training images were not available for the sampled approach used in [81]) and the fast lookup-table implementation in [82] to perform image retrieval for each of the query locations. Tab. 9 summarizes the parameters used for the experiments.

5.3 Optimistic Baselines

As explained in Sec. 5 of the paper, we implemented two *optimistic baselines*. Whereas all other localization algorithms evaluated in the paper use no prior information on a query image's pose, both optimistic baselines are given additional knowledge. For each query image, we provide a small set of reference images depicting the same part of the model. The remaining problem is to establish sufficiently many correspondences between the query and the selected reference images to facilitate camera pose estimation. Thus, both approaches measure an upper bound on the pose quality that can be

achieved with a given type of local feature.

LocalSfM. Given a query image and its relevant set of reference images, LocalSfM first extracts upright RootSIFT [48, 49] features. Next, LocalSfM performs exhaustive feature matching between the relevant reference images as well as between the query and the relevant reference images. While Active Search uses Lowe's ratio test[3], DenseSfM neither uses the ratio test nor a threshold on the maximum descriptor distance. Instead, it only requires matching features to be mutual nearest neighbors. Given the known poses and intrinsics for the reference images, LocalSfM triangulates the 3D structure of the scene using the previously established 2D-2D matches. Notice that the resulting 3D model is automatically constructed in the global coordinate system of the reference 3D model. Finally, we use the known intrinsics of the query image and the feature matches between the query and the reference images to estimate the camera pose of the query.

For each query image, the relevant set of reference images is selected as follows: For the RobotCar Seasons dataset, we use the ground truth pose of each query image to identify a relevant set of reference images. More precisely, we select all reference images whose camera centers are within 5m of the ground truth position of the query and whose orientations are within 135° of the orientation of the query image.

As explained in Sec. 3.2 of the paper, we manually select a day-time query image taken from a similar viewpoint for each nigh-time query photo in the Aachen Day-Night dataset. The day-time queries were included when constructing the intermediate model. Thus, their ground truth poses as well as a set of 3D points visible in each of them are obtain from the intermediate Structure-from-Motion model. For each day-time query, we select up to 20 reference images that observe the largest number of the 3D points visible in the day-time query. These reference images then form the set of relevant images for the corresponding night-time query photo.

LocalSfM is implemented using COLMAP [70]. It is rather straightforward to replace upright RootSIFT features with other types of local features. In order to encourage the use of our benchmark for the evaluation of local features, we will make our implementation publicly available.

DenseSfM. DenseSfM modifies the LocalSfM approach by replacing RootSIFT [48] features extracted at DoG extrema [49] with features densely extracted from a regular grid [83, 84]. The goal of this approach is to increase the robustness of feature matching between day- and night-time images [77, 85]. We used convolutional layers (conv4 and conv3) from a VGG-16 net-

[3]Active Search uses a ratio test threshold of 0.7 for 2D-to-3D and a threshold of 0.6 for 3D-to-2D matching.

Parameter	Value
Image Size	48×48 (144×48)
Patch Size	8×8
Sequence Length d_s	10

Table 10: SeqSLAM parameters.

work [75], which was pre-trained as part of the NetVLAD model (Pitts30k), as features. We generated tentative correspondences by matching the extracted features in a coarse-to-fine manner: We first match conv4 features and use the resulting matches to restrict the correspondence search for conv3 features. As for LocalSfM, we performed exhaustive pairwise image matching. The matches are verified by estimating up to two homographies between each image pair via RANSAC [71]. The resulting verified feature matches are then used as input for COLMAP [70]. The reconstruction process is the same as for LocalSfM, *i.e.*, we first triangulate the 3D points and then use them to estimate the pose of the night-time query. DenseSfM uses the same set of reference images for each query photo as LocalSfM.

SeqSLAM. We used the OpenSeqSLAM implementation from [19] with default parameters for template learning and trajectory uniqueness. For each set of synchronized Grasshopper2 images, we downscale the original 1024×1024 resolution to 48×48, then concatenate all three images to form a 144×48 pixel composite. The trajectory length parameter d_s was set to 10 images; as both the query and database images are evenly spaced this corresponds to a trajectory length of 10 meters. Tab. 10 summarizes the parameters used for the RobotCar experiments.

6 Conclusion & Lessons Learned

In this paper, we have introduced three challenging new benchmark datasets for visual localization, allowing us, for the first time, to analyze the impact of changing conditions on the accuracy of 6DOF camera pose estimation. Our experiments clearly show that the long-term visual localization problem is far from solved.

The extensive experiments performed in this paper lead to multiple interesting conclusions: (i) Structure-based methods such as Active Search and CSL are robust to most viewing conditions in urban environments. Yet, performance in the high-precision regime still needs to be improved significantly. (ii) Localizing night-time images against a database built from day-time photos is a very challenging problem, even when a location prior is given. (iii) Scenes with a significant amount of vegetation are challeng-

ing, even when a location prior is given. (iv) SfM, typically used to obtain ground truth for localization benchmarks, does not fully handle problems (ii) and (iii) due to limitations of existing local features. Dense CNN feature matching inside SfM improves pose estimation performance at high computational costs, but does not fully solve the problem. Novel (dense) features, *e.g.*, based on scene semantics [63], seems to be required to solve these problems. Our datasets readily provide a benchmark for such features through the LocalSfM and DenseSfM pipelines. (v) Image-level descriptors such as DenseVLAD can succeed in scenarios where local feature matching fails. They can even provide coarse-level pose estimates in autonomous driving scenarios. Aiming to improve pose accuracy, *e.g.*, by denser view sampling via synthetic images [22] or image-level approaches for relative pose estimation, is an interesting research direction. (vi) There is a clear benefit in using multiple images for pose estimation. Yet, there is little existing work on multi-image localization. Fully exploiting the availability of multiple images (rather than continuing the focus on single images) is thus another promising avenue for future research.

Acknowledgements. This work was partially supported by ERC grant LEAP No. 336845, CIFAR Learning in Machines & Brains program, EU-H2020 project LADIO 731970, the European Regional Development Fund under the project IMPACT (reg. no. CZ.02.1.01/0.0/0.0/15_003/0000468), JSPS KAKENHI Grant Number 15H05313, EPSRC Programme Grant EP/M019918/1, the Swedish Research Council (grant no. 2016-04445), the Swedish Foundation for Strategic Research (Semantic Mapping and Visual Navigation for Smart Robots), and Vinnova / FFI (Perceptron, grant no. 2017-01942).

Bibliography

[1] Y. Li, N. Snavely, and D. P. Huttenlocher, "Location Recognition using Prioritized Feature Matching", in *ECCV*, 2010.

[2] 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, no. 9, pp. 1744–1756, 2017.

[3] V. Larsson *et al.*, "Outlier rejection for absolute pose estimation with known orientation", in *British Machine Vision Conference*, 2016.

[4] S. Lynen *et al.*, "Get Out of My Lab: Large-scale, Real-Time Visual-Inertial Localization", in *RSS*, 2015.

[5] H. Taira *et al.*, "InLoc: Indoor Visual Localization with Dense Matching and View Synthesis", in *Conference Computer Vision and Pattern Recognition*, 2018.

[6] B. Zeisl, T. Sattler, and M. Pollefeys, "Camera pose voting for large-scale image-based localization", in *International Conference on Computer Vision*, 2015.

[7] L. Svärm *et al.*, "City-scale localization for cameras with known vertical direction", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1455–1461, 2017.

[8] T. Sattler *et al.*, "Hyperpoints and Fine Vocabularies for Large-Scale Location Recognition", in *ICCV*, 2015.

[9] Y. Li *et al.*, "Worldwide Pose Estimation Using 3D Point Clouds", in *ECCV*, 2012.

[10] S. Cao and N. Snavely, "Minimal Scene Descriptions from Structure from Motion Models", in *CVPR*, 2014.

[11] T. Sattler *et al.*, "Are Large-Scale 3D Models Really Necessary for Accurate Visual Localization?", in *CVPR*, 2017.

[12] X. Sun *et al.*, "A Dataset for Benchmarking Image-Based Localization", in *CVPR*, 2017.

[13] F. Radenović *et al.*, "From Dusk till Dawn: Modeling in the Dark", in *CVPR*, 2016.

[14] T. Sattler *et al.*, "Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions", in *CVPR*, 2018.

[15] T. Sattler *et al.*, "Image retrieval for image-based localization revisited", in *British Machine Vision Conference*, 2012.

[16] W. Maddern *et al.*, "1 Year, 1000km: The Oxford RobotCar Dataset", *IJRR*, vol. 36, no. 1, pp. 3–15, 2017.

[17] H. Badino, D. Huber, and T. Kanade, "Visual topometric localization", in *Intelligent Vehicles Symposium (IV)*, Baden-Baden, Germany, Jun. 2011.

[18] M. J. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights", in *ICRA*, 2012.

[19] N. Sünderhauf, P. Neubert, and P. Protzel, "Are we there yet? Challenging SeqSLAM on a 3000 km journey across all four seasons", in *Proc. of Workshop on Long-Term Autonomy, ICRA*, 2013.

[20] A. Torii *et al.*, "Visual Place Recognition with Repetitive Structures", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[21] Z. Chen *et al.*, "Deep Learning Features at Scale for Visual Place Recognition", *ICRA*, 2017.

[22] A. Torii *et al.*, "24/7 Place Recognition by View Synthesis", in *CVPR*, 2015.

[23] J. Shotton *et al.*, "Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images", in *CVPR*, 2013.

[24] A. Kendall, M. Grimes, and R. Cipolla, "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization", in *ICCV*, 2015.

[25] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of Michigan North Campus long-term vision and lidar dataset", *IJRR*, vol. 35, no. 9, pp. 1023–1035, 2016.

[26] D. M. Chen *et al.*, "City-Scale Landmark Identification on Mobile Devices", in *CVPR*, 2011.

[27] A. Irschara *et al.*, "From Structure-from-Motion Point Clouds to Fast Location Recognition", in *CVPR*, 2009.

[28] A. Torii *et al.*, "Visual place recognition with repetitive structures", *PAMI*, 2015.

[29] A. Geiger *et al.*, "Vision meets robotics: The KITTI dataset", *IJRR*, vol. 32, no. 11, pp. 1231–1237, 2013.

[30] S. Wang *et al.*, "TorontoCity: Seeing the World With a Million Eyes", in *ICCV*, 2017.

[31] J.-L. Blanco-Claraco, F.-Á. Moreno-Dueñas, and J. González-Jiménez, "The Málaga urban dataset: High-rate stereo and LiDAR in a realistic urban scenario", *IJRR*, vol. 33, no. 2, pp. 207–214, 2014.

[32] L. Svärm *et al.*, "City-Scale Localization for Cameras with Known Vertical Direction", *PAMI*, vol. 39, no. 7, pp. 1455–1461, 2017.

[33] B. Zeisl, T. Sattler, and M. Pollefeys, "Camera Pose Voting for Large-Scale Image-Based Localization", in *ICCV*, 2015.

[34] L. Liu, H. Li, and Y. Dai, "Efficient Global 2D-3D Matching for Camera Localization in a Large-Scale 3D Map", in *ICCV*, 2017.

[35] S. Choudhary and P. J. Narayanan, "Visibility Probability Structure from SfM Datasets and Applications", in *ECCV*, 2012.

[36] M. Donoser and D. Schmalstieg, "Discriminative Feature-to-Point Matching in Image-Based Locallization", in *Conference Computer Vision and Pattern Recognition*, 2014.

[37] F. Camposeco *et al.*, "Toroidal Constraints for Two-Point Localization under High Outlier Ratios", in *CVPR*, 2017.

[38] R. Arandjelović *et al.*, "NetVLAD: CNN architecture for weakly supervised place recognition", in *CVPR*, 2016.

[39] T. Sattler *et al.*, "Large-Scale Location Recognition and the Geometric Burstiness Problem", in *CVPR*, 2016.

[40] N. Sünderhauf *et al.*, "Place Recognition with ConvNet Landmarks: Viewpoint-Robust, Condition-Robust, Training-Free", in *RSS*, 2015.

[41] S. Lowry *et al.*, "Visual place recognition: A survey", *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.

[42] M. Cummins and P. Newman, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance", *IJRR*, vol. 27, no. 6, pp. 647–665, 2008. eprint: `http://ijr.sagepub.com/cgi/reprint/27/6/647.pdf`.

[43] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences", *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[44] R. Mur-Artal, J. M. M. Montiel, and J. D. TardÃ³s, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System", *T-RO*, vol. 31, no. 5, pp. 1147–1163, 2015.

[45] R. Arandjelović and A. Zisserman, "Dislocation: Scalable descriptor distinctiveness for location recognition", in *Asian Conference Computer Vision*, 2014.

[46] T. Naseer *et al.*, "Semantics-aware visual localization under challenging perceptual conditions", in *ICRA*, 2017.

[47] H. Jégou *et al.*, "Aggregating local descriptors into a compact image representation", in *CVPR*, Jun. 2010, pp. 3304–3311.

[48] R. Arandjelović and A. Zisserman, "Three things everyone should know to improve object retrieval", in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2012, pp. 2911–2918.

[49] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", *IJCV*, vol. 60, no. 2, 2004.

[50] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM", in *ECCV*, 2014.

[51] C. Linegar, W. Churchill, and P. Newman, "Work smart, not hard: Recalling relevant experiences for vast-scale but time-constrained localisation", in *ICRA*, 2015.

[52] C. Linegar, W. Churchill, and P. Newman, "Made to measure: Bespoke landmarks for 24-hour, all-weather localisation with a camera", in *ICRA*, 2016.

[53] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos", in *null*, IEEE, 2003, p. 1470.

[54] W. Maddern, M. Milford, and G. Wyeth, "CAT-SLAM: probabilistic localisation and mapping using a continuous appearance-based trajectory", *IJRR*, vol. 31, no. 4, pp. 429–451, 2012.

[55] T. Naseer *et al.*, "Robust visual robot localization across seasons using network flows.", in *AAAI*, 2014, pp. 2564–2570.

[56] R. Pless, "Using Many Cameras as One", in *CVPR*, 2003.

[57] G. H. Lee *et al.*, "Minimal solutions for the multi-camera pose estimation problem", *IJRR*, vol. 34, no. 7, pp. 837–848, 2015.

[58] F. Camposeco, T. Sattler, and M. Pollefeys, "Minimal Solvers for Generalized Pose and Scale Estimation from Two Rays and One Point", in *ECCV*, 2016.

[59] N. Sünderhauf *et al.*, "On the Performance of ConvNet Features for Place Recognition", in *IROS*, 2015.

[60] M. Milford *et al.*, "Sequence searching with deep-learnt depth for condition-and viewpoint-invariant route-based place recognition", in *Workshop on Computer Vision in Vehicle Technology (CVVT), CVPR*, 2015.

[61] F. Walch *et al.*, "Image-Based Localization Using LSTMs for Structured Feature Correlation", in *ICCV*, 2017.

[62] R. Clark *et al.*, "VidLoc: A Deep Spatio-Temporal Model for 6-DoF Video-Clip Relocalization", in *CVPR*, 2017.

[63] J. L. Schönberger *et al.*, "Semantic Visual Localization", in *CVPR*, 2018.

[64] P. Mühlfellner *et al.*, "Summary maps for lifelong visual localization", *Journal of Field Robotics*, 2015.

[65] S. Cao and N. Snavely, "Graph-Based Discriminative Learning for Location Recognition", in *CVPR*, 2013.

[66] P. Gronat *et al.*, "Learning and calibrating per-location classifiers for visual place recognition", *IJCV*, vol. 118, no. 3, pp. 319–336, 2016.

[67] T. Weyand, I. Kostrikov, and J. Philbin, "PlaNet - Photo Geolocation with Convolutional Neural Networks", in *ECCV*, 2016.

[68] E. Brachmann *et al.*, "Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image", in *CVPR*, 2016.

[69] E. Brachmann *et al.*, "DSAC - Differentiable RANSAC for Camera Localization", in *CVPR*, 2017.

[70] J. L. Schönberger and J.-M. Frahm, "Structure-From-Motion Revisited", in *CVPR*, 2016.

[71] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Communications of the ACM*, 1981.

[72] L. Kneip, D. Scaramuzza, and R. Siegwart, "A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation", in *CVPR*, 2011.

[73] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes", in *Robotics-DL tentative*, International Society for Optics and Photonics, 1992, pp. 586–606.

[74] R. Hartley *et al.*, "Rotation Averaging", *International Journal of Computer Vision*, vol. 103, no. 3, pp. 267–305, 2013.

[75] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv preprint arXiv:1409.1556*, 2014.

[76] J. Philbin *et al.*, "Object Retrieval with Large Vocabularies and Fast Spatial Matching", in *Conference Computer Vision and Pattern Recognition*, 2007.

[77] A. Torii *et al.*, "24/7 place recognition by view synthesis", in *CVPR*, 2015.

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

[79] D. M. Chen *et al.*, "City-scale landmark identification on mobile devices", in *Conference Computer Vision and Pattern Recognition*, 2011.

[80] A. Teynor and H. Burkhardt, "Fast codebook generation by sequential data analysis for object classification", in *International Symposium on Visual Computing*, Springer, 2007, pp. 610–620.

[81] M. Cummins and P. Newman, "Appearance-only SLAM at large scale with FAB-MAP 2.0", *IJRR*, vol. 30, no. 9, pp. 1100–1123, 2011.

[82] A. Glover *et al.*, "openFABMAP: An open source toolbox for appearance-based loop closure detection", in *ICRA*, 2012.

[83] A. Bosch, A. Zisserman, and X. Munoz, "Image classification using random forests and ferns", in *2007 IEEE 11th international conference on computer vision*, Ieee, 2007, pp. 1–8.

[84] C. Liu *et al.*, "SIFT Flow: Dense correspondence across different scenes", in *ECCV*, 2008, pp. 28–42.

[85] H. Zhou, T. Sattler, and D. W. Jacobs, "Evaluating Local Features for Day-Night Matching", in *Proc. ECCV Workshops*, 2016.

Paper II

Semantic Match Consistency for Long-Term Visual Localization

C. Toft, E. Stenborg, L. Hammarstrand, L. Brynte,
M. Pollefeys, T. Sattler and F. Kahl

European Conference on Computer Vision (ECCV) 2018.

Comment: The layout of this paper has been reformatted in order to comply with the rest of the thesis.

Semantic Match Consistency for Long-Term Visual Localization

C. Toft, E. Stenborg, L. Hammarstrand, L. Brynte, M. Pollefeys,
T. Sattler and F. Kahl

Abstract

Robust and accurate visual localization across large appearance variations due to changes in time of day, seasons, or changes of the environment is a challenging problem which is of importance to application areas such as navigation of autonomous robots. Traditional feature-based methods often struggle in these conditions due to the significant number of erroneous matches between the image and the 3D model. In this paper, we present a method for scoring the individual correspondences by exploiting semantic information about the query image and the scene. In this way, erroneous correspondences tend to get a low semantic consistency score, whereas correct correspondences tend to get a high score. By incorporating this information in a standard localization pipeline, we show that the localization performance can be significantly improved compared to the state-of-the-art, as evaluated on two challenging long-term localization benchmarks.

1 Introduction

Visual localization, i.e., estimating the camera pose of a query image with respect to a scene model, is one of the core problems in computer vision. It plays a central role in a wide range of practical applications, such as Structure-from-Motion (SfM) [1], augmented reality [2], and robotics [3], where visual navigation for autonomous vehicles has recently been receiving considerable attention.

Traditional approaches to the visual localization problem [4–10] rely on local feature descriptors to establish correspondences between 2D features found in a query image and 3D points in an SfM model of the scene. These 2D-3D matches are then used to estimate the camera pose of the query image by applying an n-point-pose solver, e.g., [11], inside a RANSAC loop [12]. While learning-based alternatives exist [13–17], they are either significantly less accurate than feature-based approaches [7, 17] or struggle to handle

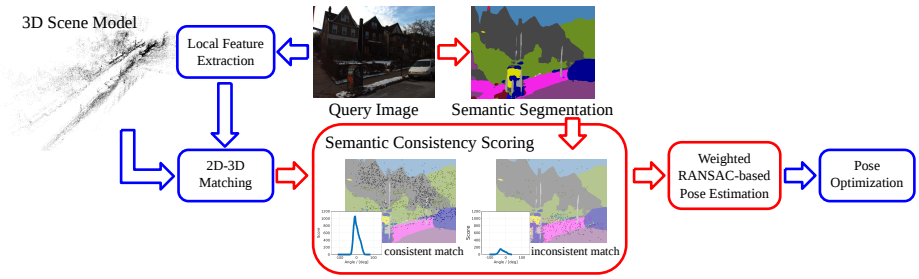


Figure 1: Illustration of the visual localization approach proposed in this paper. We extend the standard localization pipeline (blue boxes) to include a semantic consistency score (red boxes). Our approach rates the consistency of each 2D-3D match and uses the score to prioritize more consistent matches during RANSAC-based pose estimation.

larger scenes [14, 16, 18]. Feature-based approaches thus still represent the current state-of-the-art [7, 17, 18].

Existing feature-based methods for visual localization tend to work very well when the query image is taken under similar conditions as the database images used for creating the 3D model. However, feature matching performance suffers if the localization and mapping stages occur far apart in time [18], e.g., in different weather conditions, between day and night, or across different seasons. As feature detectors become less repeatable and feature descriptors less similar, localization pipelines struggle to find enough correct 2D-3D matches to facilitate successful pose estimation. One possible solution is to map the scene in as wide a range of different conditions as possible. Yet, 3D model construction and extensive data collection are costly, time-consuming, and tedious processes. At the same time, the resulting models consume a significant amount of memory. Developing localization algorithms that work well across a wide range of conditions, even if the 3D model is constructed using only a single condition, is thus desirable.

This paper presents a step towards robust algorithms for long-term visual localization through a novel strategy for robust inlier / outlier detection. The main insight is that semantic information can be used as a weak supervisory signal to distinguish between correct and incorrect correspondences: Given a semantic segmentation for each database image, we can assign a semantic label to each 3D point in the SfM model. Given a pose estimate for a query image, we can project the 3D points into a semantic segmentation of the query image. An estimate close to the correct pose should lead to a *semantically consistent* projection, where each point is projected to an image region with the same semantic label. Based on this idea, we assign each 2D-3D match a semantic consistency score, where high scores are assigned

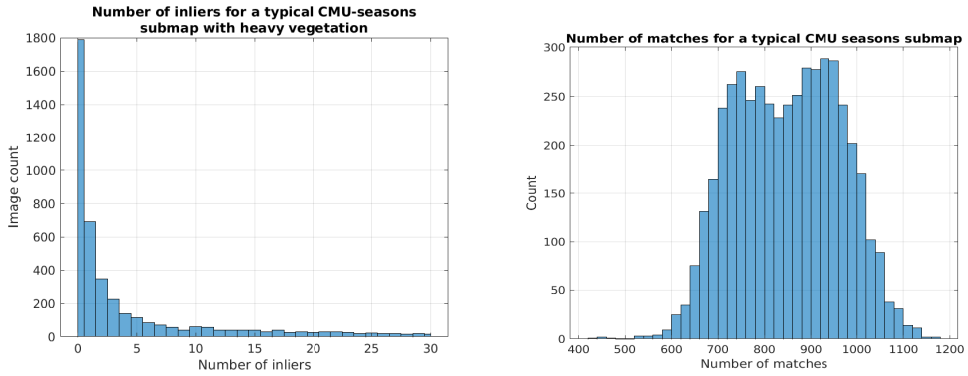


Figure 2: For each ground truth camera pose, we have counted how many true inlier 2D-3D matches have been obtained at the matching stage (shown in the left figure). Note that in around 70% of the cases, there are less than 7 true inliers. For structure-based methods, typically 12 or more consistent 2D-3D matches are required to compute and verify a camera pose, cf. [4, 5, 19]. The right figure shows a histogram over the number of matches for the same submap from the CMU Seasons dataset [18, 20].

to matches more likely to be correct. We later use these scores to bias sampling during RANSAC-based pose estimation. See Fig. 1 for an overview. While conceptually simple, this strategy leads to dramatic improvements in terms of localization rate and pose accuracy in the long-term localization scenario. The reason is that, unlike existing methods, our approach takes advantage of unmatched 3D points, and consequently copes much better with situations in which only few correct matches can be found.

While the idea of using semantics for localization is not new, cf. [21, 22], the challenge is to develop a computationally tractable framework that takes advantage of the available information. In detail, this paper makes the following contributions: 1) We present a new localization method that incorporates both standard feature matching and semantic information in a robust and efficient manner. At the center of our method is a novel semantic consistency check that allows us to rate the quality of individual 2D-3D matches. 2) We extensively evaluate and compare our method to the current state-of-the-art on two benchmarks for long-term visual localization. Our experimental results show significant improvements by incorporating semantics, particularly for challenging scenarios due to change of weather, seasonal, and lighting conditions.

The remainder of this paper is structured as follows: Sec. 2 reviews related work. Sec. 3 derives our semantic consistency score and shows how it can be incorporated into a state-of-the-art localization pipeline. Sec. 5 extensively evaluates our approach in the context of long-term localization.

2 Related Work

Traditionally, there are two approaches to solving the visual localization problem: The first one uses image retrieval techniques to find the most relevant database images for each query image [23–30]. The pose of the query image is then either approximated by the pose of the top-ranked retrieved image [24] or computed from the top-k ranking database images [7, 29, 31]. Instead of explicitly representing a scene by a database of images, another approach is to implicitly model a scene by a CNN trained for pose regression [13, 14, 17] or place classification [32].

The second approach is based on 3D scene models, typically reconstructed using SfM. Such *structure-based* methods assign one or more feature descriptors, e.g., SIFT [33] or LIFT [34], to each 3D point. For a given query image, 2D-3D correspondences are established using descriptor matching. These matches are then used to estimate the camera pose. Compared to image-retrieval approaches, structure-based methods tend to provide more accurate camera poses [7]. Yet, it is necessary that enough correct matches are found to not only estimate a pose, but also to verify that the pose is indeed correct, e.g., through inlier counting. As shown in Fig. 2 and [18], these conditions are often not satisfied when the query images are taken under significantly different conditions compared to the database images. Our approach extends structure-based methods by incorporating semantic scene understanding into the pose estimation stage.

Structure-based approaches for visual localization can be classified based on their efficiency and ability to handle more complex scenes. Approaches based on prioritized matching [4, 19, 35] focus on efficiency by terminating correspondence search once a fixed number of matches has been found. In order to handle more complex environments, robust structure-based approaches either relax the matching criteria [5, 8–10, 36] or restrict the search space [5, 6, 10, 37]. The latter type of methods use image retrieval [10, 37] or co-visibility information [5, 6] to determine which parts of the scene are visible in a query image, potentially allowing them to disambiguate matches. The former type handles the larger amount of outliers resulting from a more relaxed matching stage through deterministic outlier filtering. To this end, they use geometric reasoning to determine how consistent each match is with all other matches [8, 9, 36]. Especially when the gravity direction is known, which is typically the case in practice (e.g., via sensors or vanishing points), such approaches can handle outlier ratios of 99% or more [8, 9]. Our approach combines techniques from geometric outlier filtering [8, 9] with reasoning based on scene semantics. This enables our method to better handle scenarios where it is hard to find correct 2D-3D matches.

An alternative to obtaining 2D-3D correspondences via explicit feature matching is to directly learn the matching function [15, 16, 38–41]. Such methods implicitly represent the 3D scene structure via a random forest or CNN that predicts a 3D scene coordinate for a given image patch [39]. While these methods can achieve a higher pose accuracy than feature-based approaches [16], they also have problems handling larger outdoor scenes to the extent that training might fail completely [16, 18, 22].

The idea of using semantic scene understanding as part of the visual localization process has gained popularity over the last few years. A common strategy is to include semantics in the matching stage of visual localization pipelines, either by detecting and matching objects [42–47] or by enhancing classical feature descriptors [48–50]. The latter type of approaches still mainly relies on the strength of the original descriptor as semantics only provide a weak additional signal. Thus, these approaches do not solve the problem of finding enough correct correspondences, which motivates our work. Recent work shows that directly learning a descriptor that encodes both 3D scene geometry and semantic information significantly improves matching performance [22]. Yet, this approach requires depth maps for each query image, e.g., from stereo, which are not necessarily available in the scenario we are considering.

In contrast to the previously discussed approaches, which aim at improving the matching stage in visual localization, our method focuses on the subsequent pose estimation stage. As such, most similar to ours is existing work on semantic hypothesis verification [51] and semantic pose refinement [21]. Given a hypothesis for the alignment of two SfM models, Cohen et al. [51] project the 3D points of one model into semantic segmentations of the images used to reconstruct the other model. They count the number of 3D points projecting into regions labelled as “sky" and select the alignment hypotheses with lowest number of such free-space violations. While Cohen et al. make hard decisions, our approach avoids them by converting our semantic consistency score into sampling probabilities for RANSAC. Our approach aims at improving pose hypothesis generation while Cohen et al. only rate given hypotheses. Given an initial camera pose hypothesis, Toft et al. [21] use semantics to obtain a refined pose estimate by improving the semantic consistency of projected 3D points and curve segments. Their approach could be used as a post-processing step for the poses estimated by our method.

3 Semantic Match Consistency for Visual Localization

As outlined above, long-term localization is a hard problem due to the difficulty of establishing reliable correspondences. Our approach follows a standard feature-based pipeline and is illustrated in Fig. 1. Our central contribution is a novel semantic consistency score that is used to determine which matches are likely to be correct. Building on top of existing work on geometric outlier filtering [8, 9], we generate a set of pose hypotheses for each 2D-3D correspondence established during the descriptor matching stage. These poses are then used to measure the semantic consistency of each match. We then use the consistency scores to bias sampling inside RANSAC towards semantically consistent matches, allowing RANSAC to focus more on matches more likely to be correct.

Specifically, for each pose hypothesis generated by a given 2D-3D match, we project the visible 3D structure into the corresponding camera. Since each 3D point is endowed with a semantic label, it is possible to compare the observed semantic label in the query image with the label assigned to the 3D point. The semantic inlier count for that pose is given by the number of 3D points that project into pixels whose semantic class agrees with that of the point. The semantic consistency for the 2D-3D correspondence is then defined as the maximum semantic inlier count over all hypotheses generated by that correspondence.

Our approach offers a clear advantage over existing outlier filtering strategies [8, 9, 36]: Rather than being restricted to the 2D-3D correspondences found during the matching stage, the semantic consistency score allows us to also use unmatched 3D points when rating the 2D-3D matches. As a result, our approach is better able to handle scenarios in which it is hard to find many correct matches.

In this section, we present our proposed localization method based on semantic consistency in detail. We first introduce necessary notation. Sec. 3.1 explains the pose hypothesis generation stage. Our semantic consistency score is then described in Sec. 3.2. Finally, Sec. 3.3 summarizes the complete pipeline.

Notation. We compute the camera pose of a query image relative to a 3D point cloud that has been pre-computed using a regular Structure-from-Motion pipeline. The 3D map is defined as a set of 3D points

$$\mathcal{M} = \{(\vec{X}_i, c_i, \vec{f}_i, \vec{v}_i, \theta_i, d_i^{\text{lower}}, d_i^{\text{upper}})\}_{i=1}^{N} \ , \tag{1}$$

where N is the number of 3D points in the model. Each 3D point is defined by its 3D coordinates $\vec{X}_i$, its class label c_i (e.g., vegetation, road, *etc*),

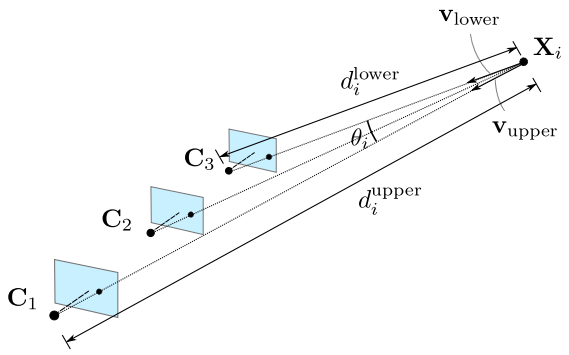


Figure 3: Example triangulation of a point during the 3D reconstruction of the point cloud. The example shows a 3D point triangulated from three observations. The quantities θ_i, d_i^{lower} and d_i^{upper} as defined in the text are shown. The vector $\vec{v}_i$ for this point is the unit vector in the middle between $\vec{v}^{\text{lower}}$ and $\vec{v}^{\text{upper}}$.

visibility information, and its corresponding (mean) feature descriptor $\vec{f}_i$. We encode the visibility information of a point as follows (cf. Fig. 3): $\vec{v}_i$ is a unit vector pointing from the 3D point towards the mean direction from which the 3D point was seen during reconstruction. It is computed by determining the two most extreme viewpoints from which the point was triangulated ($\vec{v}_{\text{lower}}$ and $\vec{v}_{\text{upper}}$ in the figure) and choosing the direction halfway between them. The angle θ_i is the angle between the two vectors. The quantities d_i^{lower} and d_i^{upper} denote the minimum and maximum distances, respectively, from which the 3D point was observed during SfM. Note that all this information is readily provided by SfM.

The semantic class labels are found by performing a pixelwise semantic labelling of all database images. For a 3D point in the SfM model, we assign its label c_i to the semantic class it was most frequently observed in.

3.1 Generating Camera Pose Hypotheses

In order to determine the semantic consistency of a single 2D-3D match $\vec{x}_j \leftrightarrow \vec{X}_j$, we compute a set of plausible camera poses for this match. We thereby follow the setup used by geometric outlier filters [8, 9] and assume that the gravity direction $\vec{g}$ in the local camera coordinates of the query image is known. This assumption is not restrictive as the gravity direction can typically be estimated very reliably from sensors or from vanishing points. In the experiments, the gravity direction in local camera coordinates was extracted from the ground truth camera pose.

Knowing the intrinsic camera calibration and the point position $\vec{X}_j$, the correspondence can be used to restrict the set of plausible camera poses

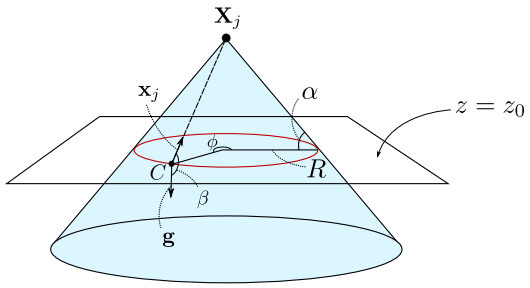


Figure 4: If the gravity direction is known in the camera's coordinate system, a single 2D-3D match $\vec{x}_j \leftrightarrow \vec{X}_j$ constrains the camera center to lie on a cone with $\vec{X}_j$ at the vertex, and whose axis is parallel to the gravity direction. If the camera height is also known, the camera center must lie on a circle (shown in red).

under which $\vec{X}_j$ exactly projects to $\vec{x}_j$ [8, 9]: The camera center must lie on a circular cone with $\vec{X}_j$ at its vertex (cf. Fig. 4). To see this, let the position of the camera center be $\vec{C}$ and the coordinates of $\vec{X}_j$ be $(x_j, y_j, z_j)^T$. In a slight abuse of notation, let $\vec{x}_j$ be the normalized viewing direction corresponding to the matching 2D feature. Since the gravity vector $\vec{g}$ in local camera coordinates is known, we can measure the angle β between the gravity direction and the line that joins $\vec{C}$ and $\vec{X}_j$ as $\beta = \arccos(\vec{g}^T \vec{x}_j)$. Assuming that the gravity direction in the 3D model coincides with the z-axis, the angle between the line joining C and $\vec{X}_j$ and the $xy-$plane then is

$$\alpha = \arccos(\vec{g}^T \vec{x}_j) - \pi/2 \ . \tag{2}$$

The set of points $\vec{C}$ such that the angle between the $xy-$plane and the line joining $\vec{C}$ and $\vec{X}_j$ equals α is a cone with $\vec{X}_j$ as the vertex. Note that the cone's position and opening angle are fully determined by $\vec{g}$ and the correspondence $\vec{x}_j \leftrightarrow \vec{X}_j$. Also note that the camera rotation is fully determined at each point of the cone [9]: two of the rotational degrees of freedom are fixed by the known gravity direction and the last degree is fixed by requiring that the viewing direction of $\vec{x}_j$ points to $\vec{X}_j$. As a result, two degrees-of-freedom remain for the camera pose, corresponding to a position on the cone's surface.

Often, the camera height z_0 can be roughly estimated from the typical depth of the 3D point in the SfM model[1]. Knowing the camera height removes one degree of freedom. As a result, the camera must lie on the circle with radius R given by $R = |z_j - z_0|/|\tan\alpha|$, which lies in the plane $z = z_0$, and whose center point is the point (x_j, y_j, z_0) [9]. For a single

[1]This strategy is used in the experiments to estimate the camera heights.

Algorithm 1 Semantic consistency score calculation for single correspondence

procedure CalculateScore($\vec{x}_j, \vec{X}_j, \vec{g}, z_0, \mathcal{M}$)
  maxScore $\leftarrow 0$
  $\alpha \leftarrow \arccos(\vec{g}^T \vec{x}_j) - \pi/2$ ▷ Angle of cone
  $R \leftarrow |z_j - z_0| / |\tan \alpha|$ ▷ Radius of circle of possible camera poses
  for $\phi \in \{0°, 1°, \ldots, 359°\}$ **do**
    score $\leftarrow 0$
    Calculate camera center $\vec{C}(\phi)$ using ϕ
    Calculate projection matrix $P(\phi)$ using R, $\vec{C}(\phi)$
    for $\vec{X}_k \in \mathcal{M}$ **do**
      $\vec{u} \leftarrow P(\phi)\vec{X}_k$ ▷ Project 3D point into query image
      if $\vec{C}(\phi) \in \mathcal{V}_k$ and $I_{\text{semantic}}(\vec{u}) = c_k$ **then**
        score $\leftarrow$ score $+ 1$ ▷ Point is visible and semantically consistent
    if score $>$ maxScore **then**
      maxScore $\leftarrow$ score
      $\vec{C}_{\text{best}} \leftarrow \vec{C}(\phi)$
  return (maxScore, $\vec{C}_{\text{best}}$)

correspondence $\vec{x}_j \leftrightarrow \vec{X}_j$, we thus generate a set of plausible camera poses by varying an angle ϕ that defines positions on this circle (cf. Fig. 4).

3.2 Measuring Semantic Match Consistency

Given a 2D-3D match $\vec{x}_j \leftrightarrow \vec{X}_j$ and its corresponding set of camera pose hypotheses (obtained by discretizing the circle into evenly spaced points), we next compute a *semantic consistency score* as described in Alg. 1.

For a camera pose hypothesis corresponding to an angle ϕ, we project the semantically labelled 3D points from the SfM model into a semantic segmentation of the query image. We then count the number of 3D points that project to a pixel whose semantic class matches that of the 3D point. For each pose on the circle, we thus find the number of 3D points that agree with the semantic labelling of the query image. The semantic consistency score for a match $\vec{x}_j \leftrightarrow \vec{X}_j$ is then defined as the maximum number of semantic inliers while sweeping the angle ϕ. Note that we project all 3D points in the model, not only the correspondences found via descriptor matching. This means that the calculation of the consistency score is not dependent of the quality of the correspondences.

Since we are using all 3D points in a model, we need to explicitly handle occlusions: a 3D point is not necessarily visible in the image even though it projects inside the image area for a given camera pose. We do so by defining a visibility volume for each 3D point from the corresponding visibility information $\vec{v}_i$, θ_i, d_i^{lower} and d_i^{upper}. The volume for the i^{th} point is defined as

$$\mathcal{V}_i = \left\{ \vec{X} \in \mathbf{R}^3 : d_i^{\text{lower}} < ||\vec{X} - \vec{X}_i|| < d_i^{\text{upper}} ||, \angle(\vec{X} - \vec{X}_i, \vec{v}_i) < \theta_i \right\} \quad . \quad (3)$$

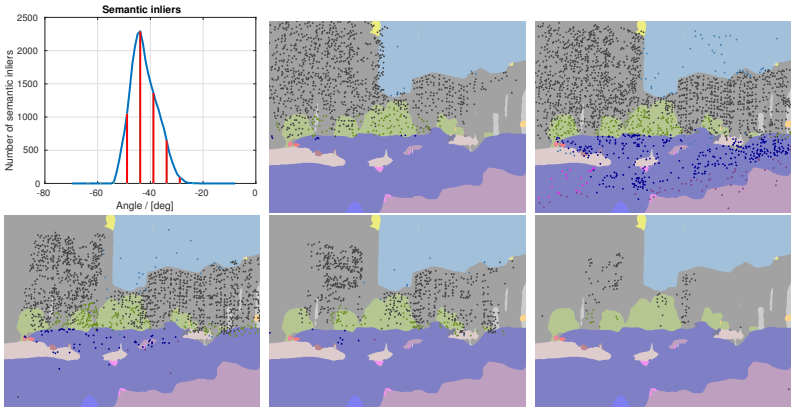


Figure 5: An example for sweeping over the angle ϕ for a single correct 2D-3D match. The upper left figure shows the number of semantic inliers as a function of ϕ. The five images with projected points correspond to the red lines in the top left and are shown in order from left to right. The upper right image corresponds to the angle that yielded the largest number of semantic inliers

A 3D point is only considered visible from a camera pose with with its center at $\vec{C}$ if $\vec{C} \in \mathcal{V}_i$. The intuition is that a 3D point only contributes to the semantic score if it is viewed from approximately the same distance and direction as the 3D point was seen from when it was triangulated during SfM. This is not too much of a restriction since local features are not completely invariant to changes in viewpoint, i.e., features naturally do not match anymore if a query image is taken too far away from the closest database image.

To further speed up the semantic scoring, we limit the set of labelled points that are projected into the image. For a 2D-3D match, only those 3D points inside a cylinder with radius R whose axis aligns with the gravity direction and goes through the 3D point $\vec{X}_j$ are considered.

Discussion. Intuitively, if a match $\vec{x}_j \leftrightarrow \vec{X}_j$ is correct, we expect the number of semantic inliers to be large for values of ϕ that correspond to camera poses close to the ground truth pose, and small for values of ϕ that yield poses distant to the ground truth pose. An example of this behavior is shown in Fig. 5. On the other hand, if a match is an outlier, we would expect only a small number of semantic inliers for all values of ϕ (cf. Fig. 1).

Naturally, the distribution of the number of semantic inliers over the angle ϕ and the absolute value of the semantic consistency score depend on how “semantically interesting" a scene is. As shown in Fig. 2, the case where many different classes are observed leads to a clear and high peak in the distribution. If only a single class is visible, e.g., “building", we can expect

a more uniform distribution, both for correct and incorrect matches. As shown later, our approach degenerates to the standard localization pipeline in this scenario.

3.3 Full Localization Pipeline

Fig. 1 shows our full localization pipeline: Given a query image, we extract local (SIFT [33]) features and compute its semantic segmentation. Using approximate nearest neighbor search, we compute 2D-3D matches between the query features and the 3D model points. We follow common practice and use Lowe's ratio test to filter out ambiguous matches [33]. Similar to work on geometric outlier filtering [8, 9, 36], we use a rather relaxed threshold of 0.9 for the ratio test to avoid rejecting correct matches. Next, we apply our proposed approach to compute a semantic consistency score per 2D-3D match (cf. Alg. 1). For each correspondence, an estimate of the camera height z_0 is obtained by checking where the database trajectory (whose poses and camera heights are available) intersects the cone of possible poses. Lastly, we apply an n-point-pose solver inside a RANSAC loop for pose estimation, using 10'000 iterations.

We use the consistency scores to adapt RANSAC's sampling scheme. More precisely, we normalize each score by the sum of the scores of all matches. We interpret this normalized score as a probability p_j and use it to bias RANSAC's sampling, i.e., RANSAC selects a match $\vec{x}_j \leftrightarrow \vec{X}_j$ with probability p_j. This can thus be seen as a "soft" version of outlier rejection: instead of explicitly removing correspondences that seem to be outliers, it just becomes unlikely that they are sampled inside RANSAC. This strategy guarantees that our approach gracefully degenerates to a standard pipeline in semantically ambiguous scenes.

4 Experimental Evaluation

In this section we present experimental evaluations of the proposed algorithm on two challenging benchmark datasets for long-term visual localization. The datasets used are the *CMU Seasons* and *RobotCar Seasons* datasets from [18].

CMU Seasons. The dataset is based on the CMU Visual Localization dataset [20]. It consists of 7,159 database images that can be used for map building, and 75,335 query images for evaluation. The images are collected from two sideways facing cameras mounted on a car while traversing the same route in Pittsburgh on 12 different occasions over the course

of a year. It captures many different environmental conditions, including overcast weather, direct sunlight, snow, and trees with and without foliage. The route contains urban and suburban areas, as well as parks mostly dominated by vegetation. All images are accompanied by accurate six degrees-of-freedom ground truth poses [18].

CMU Seasons is a very challenging dataset due to the large variations in appearance of the environment over time. Especially challenging are the areas dominated by vegetation, since these regions change drastically in appearance under different lighting conditions and in different seasons.

We used the Dilation10 network [52] trained on the Cityscapes dataset [53] to obtain the semantic segmentations. The classes used to label the 3D points were: sky, building, vegetation, road, sidewalk, pole and terrain/grass.

RobotCar Seasons. The dataset is based on a subset of the Oxford RobotCar dataset [54]. It was collected using a car-mounted camera rig consisting of 3 cameras facing to the left, right and rear of the car. The dataset consists of 26,121 database images taken at 8,707 positions, and 11,934 query images captured at 3,978 positions. All images are from a mostly urban setting in Oxford, UK, but they cover a wide variety of environmental conditions, including varying light conditions at day and night, seasonal changes from summer to winter, and various weather conditions such as sun, rain, and snow. All images have a reference pose associated with them. The average reference pose error is estimated to be below 0.10m in position and 0.5° in orientation [18].

The most challenging images of this dataset are the night images. They both exhibit a big change in lighting, but also, due to longer exposure times, contain significant motion blur.

For the RobotCar dataset, semantic segmentations were obtained using the PSPNet network [55], trained jointly on the Cityscapes [53] and Mapillary Vistas [56] datasets[2]. Additionally, 69 daytime and 13 nighttime images from the RobotCar dataset [54] were manually annotated by us, and incorporated into the training, in order to alleviate generalization issues. The classes used to label the 3D points were: sky, building, vegetation, road, sidewalk, pole and terrain/grass.

Evaluation protocol. We follow the evaluation protocol from [18], i.e., we report the percentage of query images for which the estimated pose differs by at most Xm and $Y°$ from their ground truth pose. As in [18], we use three different threshold combinations, namely (0.25m, 2°), (0.5m, 5°), and

[2]Starting from a network pretrained on Cityscapes, joint training was carried out by regarding 4 Cityscapes samples, 4 Mapillary Vistas samples and 1 RobotCar sample in each iteration. Mapillary Vistas labels were mapped to Cityscapes labels by us.

Table 1: Ablation study of our approach on the CMU Seasons dataset.

Method / Setting m deg	Urban 0.25 / 0.5 / 5 2 / 5 / 10	Suburban 0.25 / 0.5 / 5 2 / 5 / 10	Park 0.25 / 0.5 / 5 2 / 5 / 10
Weighted, P3P	75.2 / 82.1 / 87.7	44.6 / 53.9 / 63.5	30.4 / 37.8 / 48.0
Unweighted, P3P	42.5 / 50.0 / 64.5	11.5 / 16.8 / 30.1	9.3 / 13.1 / 24.2
Weighted, P2P	**81.7** / **88.0** / **92.3**	**55.4** / **65.5** / **73.1**	**39.5** / **47.5** / **58.2**
Unweighted, P2P	72.9 / 80.0 / 87.0	41.3 / 50.8 / 61.8	29.7 / 37.0 / 49.1

Table 2: Ablation study of our approach on the RobotCar Seasons dataset.

Method / Setting m deg	all day 0.25 / 0.5 / 5 2 / 5 / 10	all night 0.25 / 0.5 / 5 2 / 5 / 10
Weighted, P3P	**50.6** / **79.8** / **95.1**	7.6 / 21.5 / 45.4
Unweighted, P3P	47.0 / 74.8 / 91.9	0.5 / 4.2 / 16.0
Weighted, P2P	35.4 / 73.2 / 93.4	**13.0** / **34.1** / **63.1**
Unweighted, P2P	34.1 / 71.3 / 93.3	5.1 / 20.8 / 46.8

(5m, 10°).

4.1 Ablation Study

In this section we present an ablation study of our approach on both datasets.

The baseline is a standard, unweighted RANSAC procedure that samples each 2D-3D match with the same probability. We combine this RANSAC variant, denoted as *unweighted*, with two pose solvers: the first is a standard 3-point solver [57] (P3P) since the intrinsic calibration is known for all query images. The second solver is a 2-point solver [58] (P2P) that uses the known gravity direction. We compare both baselines against our proposed *weighted* RANSAC variant that uses our semantic consistency scores to estimate a sampling probability for each 2D-3D match. Again, we combine our approach with both pose solvers.

Tables 1 and 2 show the results of our ablation study. As can be seen, using our proposed semantic consistency scores (*weighted*) leads to clear and significant improvements in localization performance for all scenes and solvers on the CMU dataset. On the RobotCar dataset, we similar observe a significant improvement when measuring semantic consistency, with the exception of using the P2P solver during daytime. Interestingly, the P2P solver outperforms the P3P solver on both datasets using both RANSAC variants, with the exception of the daytime query images of the Oxford RobotCar dataset. The reason for this is likely due to sensitivity of the P2P solver to small noise in the ground truth vertical direction.

Table 3: Comparison of our approach, using semantic consistency scoring and the P3P pose solver, with state-of-the-art approaches on the CMU Seasons dataset.

Method / Setting m deg	Urban 0.25 / 0.5 / 5 2 / 5 / 10	Suburban 0.25 / 0.5 / 5 2 / 5 / 10	Park 0.25 / 0.5 / 5 2 / 5 / 10
ActiveSearch [19]	55.2 / 60.3 / 65.1	20.7 / 25.9 / 29.9	12.7 / 16.3 / 20.8
CSL [8]	36.7 / 42.0 / 53.1	8.6 / 11.7 / 21.1	7.0 / 9.6 / 17.0
DenseVLAD [28]	22.2 / 48.7 / 92.8	9.6 / 26.6 / **85.2**	10.3 / 27.0 / **77.0**
NetVLAD [23]	17.4 / 40.3 / **93.2**	7.7 / 20.1 / 80.5	5.6 / 15.7 / 65.8
PROSAC P3P [59]	56.7 / 64.0 / 74.2	30.6 / 38.3 / 49.1	20.0 / 25.4 / 35.1
Single-match P3P	59.3 / 66.8 / 76.2	24.6 / 32.4 / 44.6	16.8 / 22.2 / 32.6
Sem. rank. (**ours**)	**75.2** / **82.1** / 87.7	**44.6** / **53.9** / 63.5	**30.4** / **37.8** / 48.0

Table 4: Comparison of our approach, using semantic consistency scoring and the P3P pose solver, with state-of-the-art approaches on the RobotCar Seasons dataset. See the supplementary materials for a more detailed breakdown.

Method / Setting m deg	all day 0.25 / 0.5 / 5 2 / 5 / 10	all night 0.25 / 0.5 / 5 2 / 5 / 10
ActiveSearch [19]	35.6 / 67.9 / 90.4	0.9 / 2.1 / 4.3
CSL [8]	45.3 / 73.5 / 90.1	0.6 / 2.6 / 7.2
PROSAC P3P [59]	50.4 / 79.1 / 96.4	3.9 / 14.1 / 34.4
Single-match P3P	**50.7** / 79.3 / **97.2**	2.5 / 6.4 / 16.7
Sem. rank. (**ours**)	50.6 / **79.8** / 95.1	**7.6** / **21.5** / **45.4**

4.2 Comparison with State-of-the-Art

After demonstrating the benefit of our proposed semantic consistency scoring, we compare our localization pipeline against state-of-the-art approaches on both datasets, using the results reported in [18]. More concretely, we compare against ActiveSearch (AS) [19] and the City-Scale Localization (CSL) [8] methods, which represent the state-of-the-art in efficient and scalable localization, respectively. In addition, we compare against two image retrieval-based baselines, namely DenseVLAD [28] and NetVLAD [23], when their results are available in [18]. We omitted results for the methods LocalSfM, DenseSfM, ActiveSearch+Generalized Camera, and FABMAP [60] present in [18], since these use either a sequence of images (the latter two), costly SfM approaches coupled with a strong location prior (the former two), or use ground truth information (the former three), and are thus not directly comparable. For a fair comparison with AS and CSL, we use the variant of our localization pipeline that uses semantic consistency scoring and the P3P solver.

Tables 3 and 4 show the results of our comparison. As can be seen, our approach significantly outperforms both AS and CSL, especially in the

high-precision regime. Especially the comparison with CSL is interesting as our pose generation stage is based on its geometric outlier filtering strategy. The clear improvements over CSL validate our idea of incorporating scene semantics into the pose estimation stage in general and the idea of using non-matching 3D points to score matches in particular.

On the CMU dataset, both DenseVLAD and NetVLAD can localize more query images in the coarse-precision regime (5 m, 10°). Both approaches represent images using a compact image-level descriptor and approximate the pose of the query image using the pose of the top-retrieved database image. Both methods do not use any feature matching between images. As shown in Fig. 6, this allows DenseVLAD and NetVLAD to handle scenarios with very strong appearance changes in which feature matching completely fails. Note that both DenseVLAD or NetVLAD could be used as a fallback option for our approach.

Interestingly, the P3P RANSAC baseline outperforms AS and CSL in several instances. This is likely due to differing feature matching strategies and different numbers of RANSAC iterations. Active Search uses a very strict ratio test, which causes problems in challenging scenes. CSL was evaluated on CMU Seasons by keeping all detected features (no ratio test), resulting in several thousand matches per image. CSL may have yielded better results with a ratio test.

In addition, we also compare our approach to two methods based on P3P RANSAC. The first is PROSAC [59], a RANSAC variant that uses a deterministic sampling strategy, where correspondences deemed more likely to be correct are given higher priority during sampling. In our experiments, the quality measure used was the Euclidean distance between the descriptors of the observed 2D point and the corresponding matched 3D point.

The second RANSAC variant employs a very simple single-match semantic outlier rejection strategy: all 2D-3D matches for which the semantic labels of the 2D feature and 3D point do not match are discarded before pose estimation.

As can be seen in Tables 3 and 4, all three methods perform similarly well on the relatively easy daytime queries of the RobotCar Seasons dataset. However, our approach significantly outperforms the other two methods under all other conditions. This clearly validates our idea of semantic consistency scoring.

5 Conclusion

In this paper, we have presented a method for soft outlier filtering by using the semantic content of a query image. Our method ranks the 2D-3D

Figure 6: Illustrations of the result of our method on the CMU Seasons dataset. Rows 1 and 3 show query images that our method successfully localizes (error $< .25$ m) while DenseVLAD and AS fail (error > 10 m) and rows 2 and 4 the vice versa. Green boxes indicate true correspondences, while gray circles indicate false correspondences. White/red crosses indicate correctly/incorrectly detected inliers, respectively.

matches found by feature-based localization pipelines depending on how well they agree with the scene semantics. Provided that the gravity direction and camera height are (roughly) known, the camera is constrained to lie on a circle for a given match. Traversing this circle and projecting the semantically labelled scene geometry into the query image, we calculate a semantic consistency score for this match based on the fit between the projected and observed semantic labels. The scores are then used to bias sampling during RANSAC-based pose estimation.

Experiments on two challenging benchmarks for long-term visual localization show that our approach outperforms state-of-the-art methods. This validates our idea of using scene semantics to distinguish correct and wrong matches and shows the usefulness of semantic information in the context of visual localization.

Acknowledgements This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by

the Knut and Alice Wallenberg Foundation, the Swedish Research Council (grant no. 2016-04445), the Swedish Foundation for Strategic Research (Semantic Mapping and Visual Navigation for Smart Robots), and Vinnova /FFI (Perceptron, grant no. 2017-01942).

Bibliography

[1] J. L. Schönberger and J.-M. Frahm, "Structure-From-Motion Revisited", in *CVPR*, 2016.

[2] R. O. Castle, G. Klein, and D. W. Murray, "Video-rate localization in multiple maps for wearable augmented reality", in *ISWC*, 2008.

[3] S. Lynen *et al.*, "Get Out of My Lab: Large-scale, Real-Time Visual-Inertial Localization", in *RSS*, 2015.

[4] Y. Li, N. Snavely, and D. P. Huttenlocher, "Location Recognition using Prioritized Feature Matching", in *ECCV*, 2010.

[5] Y. Li *et al.*, "Worldwide Pose Estimation Using 3D Point Clouds", in *ECCV*, 2012.

[6] L. Liu, H. Li, and Y. Dai, "Efficient Global 2D-3D Matching for Camera Localization in a Large-Scale 3D Map", in *ICCV*, 2017.

[7] T. Sattler *et al.*, "Are Large-Scale 3D Models Really Necessary for Accurate Visual Localization?", in *CVPR*, 2017.

[8] L. Svärm *et al.*, "City-Scale Localization for Cameras with Known Vertical Direction", *PAMI*, vol. 39, no. 7, pp. 1455–1461, 2017.

[9] B. Zeisl, T. Sattler, and M. Pollefeys, "Camera Pose Voting for Large-Scale Image-Based Localization", in *ICCV*, 2015.

[10] T. Sattler *et al.*, "Hyperpoints and Fine Vocabularies for Large-Scale Location Recognition", in *ICCV*, 2015.

[11] L. Kneip, D. Scaramuzza, and R. Siegwart, "A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation", in *CVPR*, 2011.

[12] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Communications of the ACM*, 1981.

[13] A. Kendall, M. Grimes, and R. Cipolla, "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization", in *ICCV*, 2015.

[14] A. Kendall and R. Cipolla, "Geometric loss functions for camera pose regression with deep learning", in *CVPR*, 2017.

[15] E. Brachmann *et al.*, "DSAC - Differentiable RANSAC for Camera Localization", in *CVPR*, 2017.

[16] E. Brachmann and C. Rother, "Learning Less is More - 6D Camera Localization via 3D Surface Regression", in *CVPR*, 2018.

[17] F. Walch *et al.*, "Image-Based Localization Using LSTMs for Structured Feature Correlation", in *ICCV*, 2017.

[18] T. Sattler *et al.*, "Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions", in *CVPR*, 2018.

[19] T. Sattler, B. Leibe, and L. Kobbelt, "Efficient & Effective Prioritized Matching for Large-Scale Image-Based Localization", *PAMI*, vol. 39, no. 9, pp. 1744–1756, 2017.

[20] H. Badino, D. Huber, and T. Kanade, "Visual topometric localization", in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, IEEE, 2011, pp. 794–799.

[21] C. Toft, C. Olsson, and F. Kahl, "Long-term 3D Localization and Pose from Semantic Labellings", in *ICCV Workshops*, 2017.

[22] J. L. Schönberger *et al.*, "Semantic Visual Localization", in *CVPR*, 2018.

[23] R. Arandjelović *et al.*, "NetVLAD: CNN architecture for weakly supervised place recognition", in *CVPR*, 2016.

[24] D. M. Chen *et al.*, "City-Scale Landmark Identification on Mobile Devices", in *CVPR*, 2011.

[25] J. Knopp, J. Sivic, and T. Pajdla, "Avoding Confusing Features in Place Recognition", in *ECCV*, 2010.

[26] T. Sattler *et al.*, "Large-Scale Location Recognition and the Geometric Burstiness Problem", in *CVPR*, 2016.

[27] G. Schindler, M. Brown, and R. Szeliski, "City-Scale Location Recognition", in *CVPR*, 2007.

[28] A. Torii *et al.*, "24/7 Place Recognition by View Synthesis", in *CVPR*, 2015.

[29] A. R. Zamir and M. Shah, "Accurate Image Localization Based on Google Maps Street View", in *ECCV*, 2010.

[30] A. R. Zamir and M. Shah, "Image Geo-Localization Based on MultipleNearest Neighbor Feature Matching Using Generalized Graphs", *PAMI*, vol. 36, no. 8, pp. 1546–1558, 2014.

[31] W. Zhang and J. Kosecka, "Image based Localization in Urban Environments", in *3DPVT*, 2006.

[32] T. Weyand, I. Kostrikov, and J. Philbin, "PlaNet - Photo Geolocation with Convolutional Neural Networks", in *ECCV*, 2016.

[33] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", *IJCV*, vol. 60, no. 2, 2004.

[34] K. M. Yi *et al.*, "LIFT: Learned Invariant Feature Transform", in *ECCV*, 2016.

[35] S. Choudhary and P. J. Narayanan, "Visibility Probability Structure from SfM Datasets and Applications", in *ECCV*, 2012.

[36] F. Camposeco *et al.*, "Toroidal Constraints for Two-Point Localization under High Outlier Ratios", in *CVPR*, 2017.

[37] A. Irschara *et al.*, "From Structure-from-Motion Point Clouds to Fast Location Recognition", in *CVPR*, 2009.

[38] T. Cavallari *et al.*, "On-The-Fly Adaptation of Regression Forests for Online Camera Relocalisation", in *CVPR*, 2017.

[39] D. Massiceti *et al.*, "Random Forests versus Neural Networks - What's Best for Camera Relocalization?", in *ICRA*, 2017.

[40] J. Shotton *et al.*, "Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images", in *CVPR*, 2013.

[41] J. Valentin *et al.*, "Exploiting Uncertainty in Regression Forests for Accurate Camera Relocalization", in *CVPR*, 2015.

[42] S. Ardeshir *et al.*, "GIS-Assisted Object Detection and Geospatial Localization", in *ECCV*, 2014.

[43] N. Atanasov *et al.*, "Localization from semantic observations via the matrix permanent", *IJRR*, vol. 35, no. 1-3, pp. 73–99, 2016.

[44] A. Cohen *et al.*, "Indoor-Outdoor 3D Reconstruction Alignment", in *ECCV*, 2016.

[45] R. F. Salas-Moreno *et al.*, "SLAM++: Simultaneous Localisation and Mapping at the Level of Objects", in *CVPR*, 2013.

[46] M. Schreiber, C. Knöppel, and U. Franke, "LaneLoc: Lane marking based localization using highly accurate maps", in *IV*, 2013.

[47] F. Yu, J. Xiao, and T. A. Funkhouser, "Semantic alignment of LiDAR data at city scale", in *CVPR*, 2015.

[48] R. Arandjelović and A. Zisserman, "Visual Vocabulary with a Semantic Twist", in *ACCV*, 2014.

[49] N. Kobyshev, H. Riemenschneider, and L. V. Gool, "Matching Features Correctly through Semantic Understanding", in *3DV*, 2014.

[50] G. Singh and J. Košecká, "Semantically Guided Geo-location and Modeling in Urban Environments", in *Large-Scale Visual Geo-Localization*, 2016.

[51] A. Cohen, T. Sattler, and M. Pollefeys, "Merging the Unmatchable: Stitching Visually Disconnected SfM Models", in *ICCV*, 2015.

[52] F. Yu and V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions", in *ICLR*, 2016.

[53] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding", in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[54] W. Maddern *et al.*, "1 Year, 1000km: The Oxford RobotCar Dataset", *IJRR*, vol. 36, no. 1, pp. 3–15, 2017.

[55] H. Zhao *et al.*, "Pyramid Scene Parsing Network", in *CVPR*, 2017.

[56] G. Neuhold *et al.*, "The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes", in *ICCV*, 2017.

[57] R. Haralick *et al.*, "Review and analysis of solutions of the three point perspective pose estimation problem", *IJCV*, vol. 13, no. 3, pp. 331–356, 1994.

[58] Z. Kukelova, M. Bujnak, and T. Pajdla, "Closed-form Solutions to Minimal Absolute Pose Problems with Known Vertical Direction", in *ACCV*, 2011.

[59] O. Chum and J. Matas, "Matching with PROSAC - progressive sample consensus", in *CVPR*, 2005.

[60] M. Cummins and P. Newman, "Appearance-only SLAM at large scale with FAB-MAP 2.0", *IJRR*, vol. 30, no. 9, pp. 1100–1123, 2011.

Supplementary Materials

This supplementary material presents additional material not included in the main paper: Sec. 6 shows more detailed results for the RobotCar Seasons dataset. Sec. 7 shows example images for the RobotCar Seasons dataset. Finally, Sec. 8 provides information about the run time of the method.

6 Detailed Results for the RobotCar Seasons Dataset

In the main article, we provided localization results for the *day-all* and *night-all* conditions of the RobotCar Seasons dataset [18, 54]. Here, we present a more detailed breakdown of the day and night conditions into the different sub-conditions defined in [18]. Due to the large size of the table, we have divided it into two tables, Tab. 5 and 6. The different conditions are: *Dawn*, *Dusk*, *Overcast-summer*, *Overcast-winter*, *Rain*, *Snow*, *Sun*, *Dawn*, *Night* and *Night-rain*. The last two make up the *night-all* category in the main article, and the rest make up the *day-all* category.

Note that for most day conditions (when good correspondences are generally present), performing the semantic consistency ranking gives no significant increase in performance. For the more challenging conditions (such as *Sun*, *Night* and *Night-rain*), ranking the correspondences based on their semantic consistency allows the RANSAC procedure to find a better inlier set by making it more unlikely to pick outlier correspondences. For these conditions, we observe a significant improvement in localization performance for our approach.

Table 5: Additional localization results on the Oxford Seasons dataset, showing results for conditions *Dawn*, *Dusk*, *Overcast-summer*, *Overcast-winter* and *Rain*. Results from the reference methods are taken from the benchmark article [18].

Method m deg	Dawn 0.25 / 0.5 / 5.0 2 / 5 / 10	Dusk 0.25 / 0.5 / 5.0 2 / 5 / 10	OC-summer 0.25 / 0.5 / 5.0 2 / 5 / 10	OC-winter 0.25 / 0.5 / 5.0 2 / 5 / 10	Rain 0.25 / 0.5 / 5.0 2 / 5 / 10
ActiveSearch [19]	36.2 / 68.7 / 89.4	44.7 / 74.6 / 95.9	24.8 / 63.9 / 95.5	33.1 / 71.5 / 93.8	51.3 / 79.8 / 96.9
CSL [8]	47.2 / 73.3 / 90.1	56.6 / 82.7 / 95.9	34.1 / 71.1 / 93.5	39.5 / 75.9 / 92.3	59.6 / 83.1 / 97.6
DenseVLAD [28]	8.9 / 36.9 / 92.5	10.2 / 38.8 / 94.2	6.0 / 29.8 / 92.0	4.4 / 26.7 / 93.3	10.2 / 40.6 / 96.9
NetVLAD [23]	6.2 / 22.8 / 82.6	7.4 / 29.7 / 92.9	6.5 / 29.6 / 95.2	3.1 / 25.9 / 92.6	9.0 / 35.9 / 96.0
PROSAC [59]	53.6 / 79.9 / 94.4	55.3 / 83.2 / 95.9	40.6 / 76.5 / 99.1	43.1 / 78.7 / 97.4	61.2 / 82.1 / 98.1
Single-match	53.8 / 80.5 / 95.5	57.1 / 82.5 / 97.5	37.6 / 75.4 / 98.3	43.3 / 78.7 / 97.4	62.5 / 82.7 / 98.8
Weighted, P3P	53.4 / 81.0 / 97.1	53.8 / 83.0 / 97.7	39.5 / 75.6 / 92.4	39.5 / 72.3 / 85.1	62.0 / 82.4 / 99.0
Unweighted, P3P	52.4 / 77.4 / 95.4	58.9 / 83.8 / 97.7	36.7 / 69.3 / 89.2	36.2 / 70.3 / 81.3	61.8 / 82.9 / 98.8
Weighted, P2P	47.4 / 79.5 / 94.8	47.5 / 81 / 95.9	21.6 / 66.1 / 91.6	32.3 / 66.9 / 85.1	31.1 / 74.8 / 95.2
Unweighted, P2P	48.2 / 79.1 / 94.2	44.7 / 82.2 / 95.4	18.6 / 60.5 / 91.8	30.8 / 65.1 / 85.1	30.4 / 75.0 / 94.8

Table 6: Additional localization results on the Oxford Seasons dataset, showing results for conditions *Snow*, *Sun*, *Night*, and *Night-rain*. Results from the reference methods are taken from the benchmark article [18].

Method m deg	Snow 0.25 / 0.5 / 5.0 2 / 5 / 10	Sun 0.25 / 0.5 / 5.0 2 / 5 / 10	Night 0.25 / 0.5 / 5.0 2 / 5 / 10	Night-rain 0.25 / 0.5 / 5.0 2 / 5 / 10
ActiveSearch [19]	36.4 / 72.2 / 93.7	25.0 / 46.5 / 69.1	0.5 / 1.1 / 3.4	1.4 / 3.0 / 5.2
CSL [8]	53.2 / 83.6 / 92.4	28.0 / 47.0 / 70.4	0.2 / 0.9 / 5.3	0.9 / 4.3 / 9.1
DenseVLAD [28]	8.6 / 30.1 / 90.2	5.7 / 16.3 / 80.2	0.9 / 3.4 / 19.9	1.1 / 5.5 / 25.5
NetVLAD [23]	7.0 / 25.2 / 91.8	5.7 / 16.5 / 86.7	0.2 / 1.8 / 15.5	0.5 / 2.7 / 16.4
PROSAC [59]	56.6 / 85.9 / 96.7	41.2 / 68.0 / 93.3	3.2 / 8.9 / 29.2	4.5 / 19.3 / 39.5
Single-match	58.1 / 86.1 / 97.1	42.6 / 69.6 / 95.2	2.7 / 6.8 / 18.5	2.2 / 6.0 / 15
Weighted, P3P	56.4 / 85.5 / 98	46.5 / 74.6 / 95.9	6.2 / 18.5 / 44.3	8.0 / 26.4 / 46.4
Unweighted, P3P	54.8 / 85.5 / 96.9	29.6 / 54.8 / 83.5	0.2 / 4.1 / 15.8	0.7 / 4.3 / 16.4
Weighted, P2P	46 / 81.4 / 96.3	21.7 / 62.6 / 94.1	10 / 25.8 / 61	15.9 / 42.3 / 65.2
Unweighted, P2P	44.8 / 80.8 / 96.3	20.7 / 56.1 / 94.3	4.1 / 16.0 / 44.7	6.1 / 25.7 / 48.9

7 RobotCar Seasons examples

Most of the daytime images from the Oxford seasons data set are fairly easy to localize correctly due to an abundance of buildings in the images. The visual appearance of these buildings stays fairly constant, and these buildings thus provide good, stable interest points to localize with. Most failure cases can be found in the nighttime images. Fig. 7 shows examples for these failure cases. We can see that the semantic classification fails for large parts of the nighttime images, as buildings and even sky are mislabelled. However, this is not particularly surprising given the limited amount of nighttime training examples that the semantic segmentation algorithm has seen during training.

8 Timing

In this section we present some information about the runtime of the presented algorithm. Fig. 8 shows histograms over the time required to calculate the semantic consistency score per correspondence for all images in the CMU Seasons dataset as well as the RobotCar Seasons dataset. Note that the semantic scoring is perfectly parallel: the scores can be calculated completely independently of one another. The algorithm is thus very well suited for a parallel implementation. The histograms shows the time taken for an unoptimized MATLAB implementation of the algorithm to calculate the semantic consistency score for one correspondence.

Since the calculation of the consistency score mostly requires matrix-vector multiplications (projections and angle calculations), the algorithm could, due to its parallel nature, be implemented on a GPU for a significant speedup if desired.

The time taken to score RobotCar Seasons correspondences is in gen-

eral higher than for CMU Seasons correspondences since more points are generally visible at each camera position.

In our implementation, the most time-consuming part is to check which points are visible from each camera position, i.e., to check whether $\vec{C} \in \mathcal{V}_i$, for each i. This part of our approach could be accelerated by pre-computing a covisibility graph for the 3D points in the map.

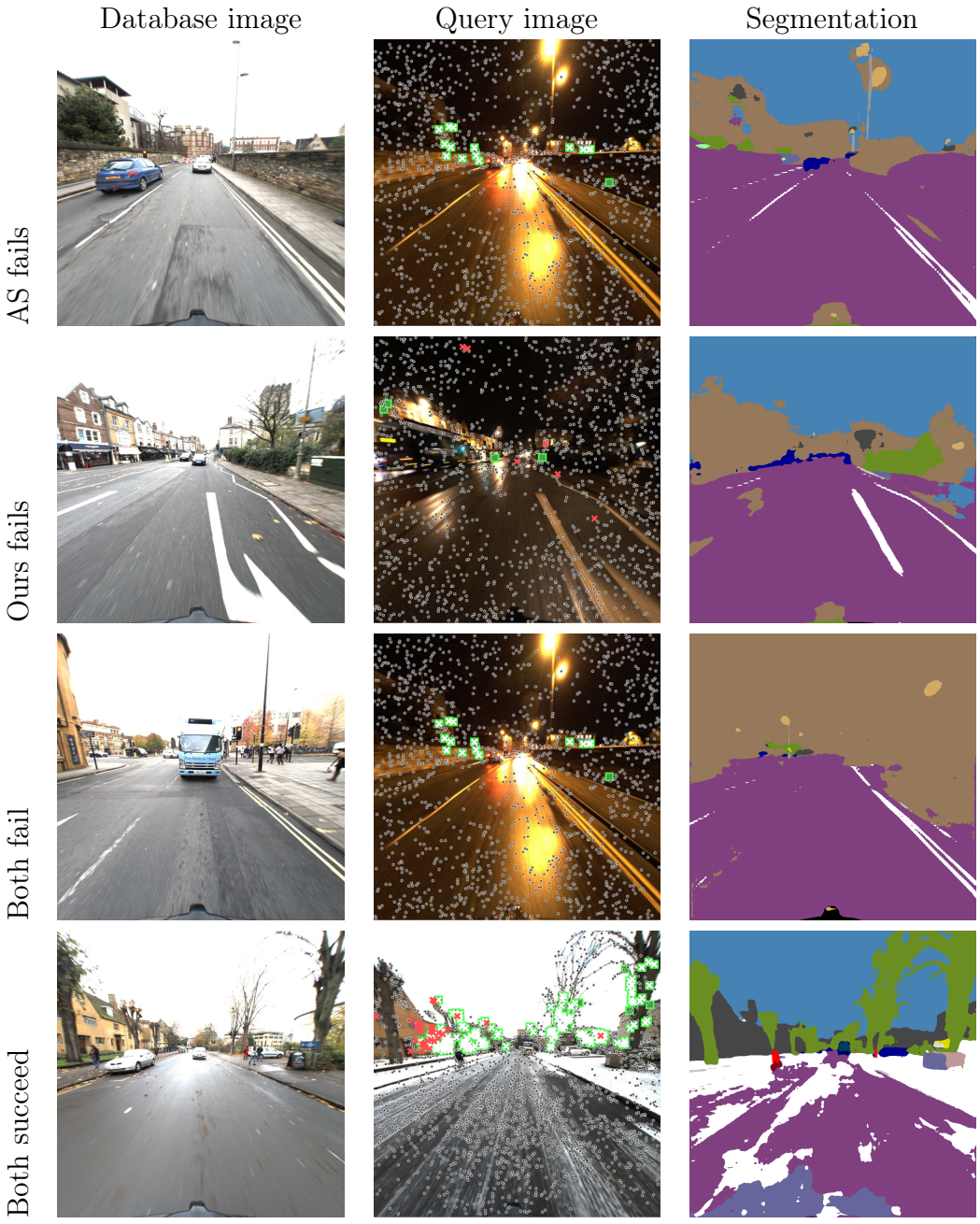


Figure 7: Illustrations of the result of our method on the RobotCar Seasons dataset. Row 1 shows an example where Active Search fails, but our method succeeds. Row 2 shows an example of the opposite case. Here we can notice that all four correct correspondences are on the buildings, but we also see that buildings have been misclassified in the segmentation (they should be gray). The two bottom rows show examples where both algorithms perform similarly. Left: Example images used to construct the database model. Middle: Query images with feature correspondences. Green boxes indicate true correspondences, while gray circles indicate false correspondences. White/red crosses indicate correctly/incorrectly detected inliers, respectively. Right: Semantic segmentations of the query images.

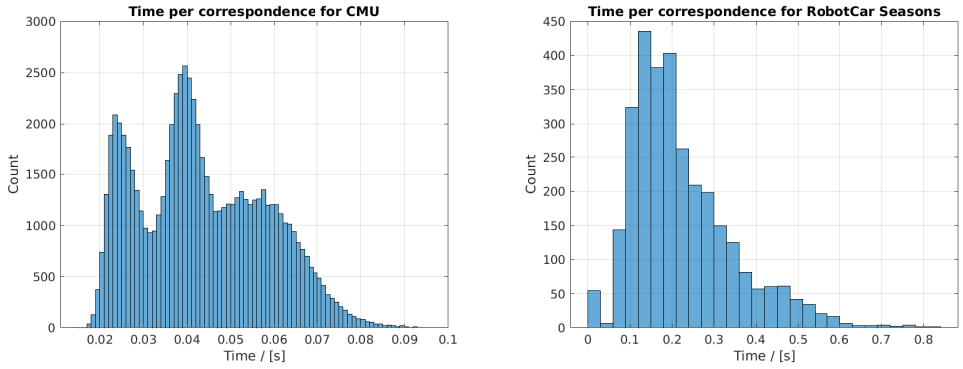


Figure 8: Histogram over the time required to calculate the semantic consistency score per correspondence for all images in the CMU Seasons and the RobotCar Seasons datasets.

Paper III

Long-term Visual Localization Using Semantically Segmented Images

E. Stenborg, C. Toft and L. Hammarstrand

International Conference on Robotics and Automation (ICRA) 2018

Comment: The layout of this paper has been reformatted in order to comply with the rest of the thesis.

Long-term Visual Localization Using Semantically Segmented Images

E. Stenborg, C. Toft and L. Hammarstrand

Abstract

Robust cross-seasonal localization is one of the major challenges in long-term visual navigation of autonomous vehicles. In this paper, we exploit recent advances in semantic segmentation of images, i.e., where each pixel is assigned a label related to the type of object it represents, to solve the problem of long-term visual localization. We show that semantically labeled 3D point maps of the environment, together with semantically segmented images, can be efficiently used for vehicle localization without the need for detailed feature descriptors (SIFT, SURF, etc.). Thus, instead of depending on hand-crafted feature descriptors, we rely on the training of an image segmenter. The resulting map takes up much less storage space compared to a traditional descriptor based map. A particle filter based semantic localization solution is compared to one based on SIFT-features, and even with large seasonal variations over the year we perform on par with the larger and more descriptive SIFT-features, and are able to localize with an error below 1 m most of the time.

1 Introduction

Although autonomous vehicle navigation can be done in uncharted environments, most efforts aiming at self-driving vehicles usable for every day activities, such as commuting, rely on pre-constructed maps to provide information about drivable road ahead. A central task for the self-driving vehicle is then to find its current location in these maps using observations from its on-board sensors, such as camera, lidar, radar etc. For this, in addition to navigational information, the maps typically describe the position of landmarks, i.e., points or structures in the environment, that can easily be detected by the on-board sensors. When it comes to cameras, it is common to use point features in the images as landmarks. The associated map is then constructed from these point features, where each feature is described by its 3D position in the world and a condensed description of the visual appearance of the local neighborhood around the feature. In

the localization phase, these descriptors are used to find correspondences between point features in the current image and the features in the map [1, 2]. A variety of methods for establishing these 2D-3D correspondences have been investigated, and once found, they can be used for calculating the full six-degrees-of-freedom camera pose [3–5].

The visual information captured by a camera is well suited for most tasks related to driving. If interpreted correctly, it can be used, e.g., to detect other road users or drivable road surface, and also to localize ourselves. However, the abundance of information also provides several difficult challenges. For example, the appearance of our feature points may change due to changes in light, weather, and seasonal variations. The traditional point descriptors used, e.g., SIFT, SURF, BRIEF, have been carefully designed to be robust towards uniform intensity changes and slight variation of viewpoint, but most were not designed to be invariant against large changes in lighting (day/night) or the fact that a tree looks completely different in summer compared to in winter. Additionally, it has been shown that the most commonly used feature detectors are very sensitive to changes in lighting conditions[6], implying that even if the feature descriptor is robust to these environmental changes, the resulting feature matches would still be incorrect since the detector does not trigger at the same points during localization as during mapping. Thus, when mapping and localization occur in sufficiently dissimilar conditions, it is very difficult to reliably match features between the image and the map, resulting in poor positioning accuracy or even complete failure of the localization algorithm. This long-term localization problem typically gets harder as the map gets older [7], and is one of the major challenges in long-term autonomy.

The problem can be boiled down to finding a description of the environment that is both usable for localization, invariant over time, and compact. If this can not be achieved, one has to cope with changing conditions by continuously updating the map [7]. An attempt at having more robust feature points and descriptors is presented in [8], where they, instead of using handcrafted feature descriptors, train neural networks to produce more robust feature descriptors. Although they show promising results compared to SIFT they are not designed to handle the type of variations described above.

In this paper we propose to use recent advances in semantic segmentation of images [9], and design a localization algorithm based on these semantically segmented images and a semantic point feature map, where, instead of using the traditional descriptors to describe our features, each point is only described by the semantic class of the object on which it resides. By semantic class, we mean a classification into a a few classes that

are meaningful for a human, e.g. "road", "building", "vegetation", etc. The seasonal invariance is thus off-loaded from the feature descriptors to the semantic segmentation algorithm. The aim is to show that localization works well when the semantic classification is reasonably correct despite the more space efficient representation of the environment. The positioning performance of the proposed algorithm is compared to a localization algorithm based on a traditional SIFT point feature map, using data collected throughout a year.

2 Problem statement

This paper concerns the problem of sequentially finding the current position of a vehicle in a point feature map using on-board cameras. The dataset considered here comes from Carnegie Mellon University [10], and contains both video, GPS measurements and odometry. In this section we present the available observations in more detail and introduce notation for the map. We conclude by defining the problem at hand mathematically.

2.1 Observations

The observations are taken with irregular, but known time intervals. We denote the time instance for which one such measurement was taken as t. Below follows a description of the information coming from the sensors at one of these time instances. Relevant coordinate frames and mounting positions are depicted in Fig. 1.

Odometry

The odometry provides a 3-D velocity and 3-D rotational velocity, denoted $\mathbf{v}_t = [v_t^x, v_t^y, v_t^z]^T$ and $\boldsymbol{\omega}_t = [\omega_t^z, \omega_t^y, \omega_t^x]^T$, respectively. The velocities are given relative the vehicle frame, and the superscripts indicate along or around which axis the component acts.

Images

The vehicle is equipped with a pair of calibrated cameras, mounted as indicated in Fig. 1. At a frequency of about 15 Hz each camera takes an RGB image with resolution 1024×768. Although it is possible to use this raw image data directly [11], it is somewhat complicated. A more common approach is to condense the image into a set of feature points with associated descriptor vector, and view this as the measurement. As such, the image is pre-processed to produce a set of n_t feature points and descriptor

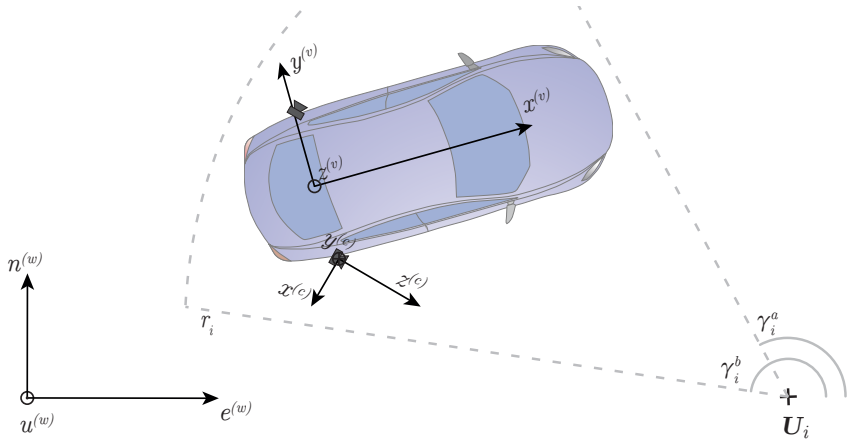


Figure 1: Coordinate frames used are "world" in local ENU (w), vehicle (v), and camera (c). The origin of the vehicle frame is taken to be the mid point between the cameras, which gives us a horizontal lever arm, perpendicular to the direction of travel, to each camera. Also, the visibility wedge $(\gamma_i^a, \gamma_i^b, r_i)$ of a map point, $\mathbf{U}_i$, is illustrated.

pairs, $\mathbf{f}_t = \{\langle \mathbf{u}_t^i, \mathbf{d}_t^i \rangle\}_{i=1}^{n_t}$, where $\mathbf{u}_t^i$ is a normalized image coordinate and $\mathbf{d}_t^i$ the associated descriptor vector.

In this paper, $\mathbf{f}_t$ will have different properties depending on which map we are using. In our proposed method (semantic point feature map) $\mathbf{f}_t$ will be dense and contain an element for each pixel in the image, see Fig. 2. In the case of a SIFT based map on the other hand, $\mathbf{f}_t$ is sparse and contains only the pixels for which the SIFT-algorithm has generated a detection and their associated SIFT-descriptors (a 128x1 vector).

2.2 Maps

We assume that we have a pre-constructed point feature map consisting of M point features. Let us denote the map $\mathcal{M} = \{\langle \mathbf{U}_i, \mathbf{D}_i, \mathcal{V}_i \rangle\}_{i=1}^{M}$. Each point feature is described by its global position $\mathbf{U}_i = [U_i^e, U_i^n, U_i^u]^T$ (east, north and up, respectively), its associated descriptor vector, $\mathbf{D}_i$ and visibility $\mathcal{V}_i = [\rho_i, \gamma_i^a, \gamma_i^b, r_i]^T$. The visibility of a feature point is parameterized by a probability of detection ρ_i and a visibility volume defined by γ_i^a, γ_i^b, and r_i. The i^{th} point is modeled to have a detection probability of ρ_i in the wedge shaped volume defined by the two angles γ_i^a, and γ_i^b, in the horizontal plane, out to a range, r_i, from the point, and 0 elsewhere, see Fig. 1.

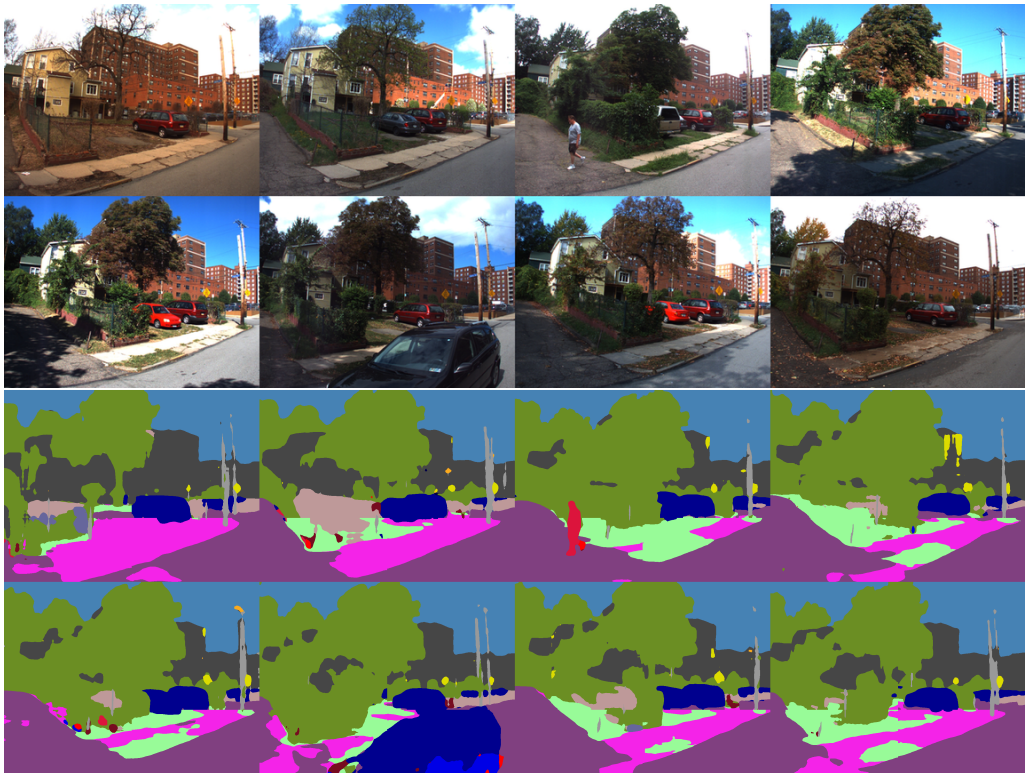


Figure 2: Example of how visual appearance changes with time for a scene (top half), and that semantic segmentation of the same images (bottom half) show less variation over time, although there are still large areas, especially around the tree and on the sidewalk which are misclassified.

2.3 Problem definition

The problem at hand is to recursively calculate the posterior density of the pose of the host vehicle relative to a map, $\mathcal{M}$, given all observations. That is, assuming that the pose of the host vehicle at time t is described by the state $\mathbf{x}_t$, we want to sequentially calculate the density $p(\mathbf{x}_t|\mathbf{f}_{1:t}, \mathcal{M})$. Further, in this paper we assume that the vehicle state is given as $\mathbf{x}_t = [e_t, n_t, u_t, \gamma_t, \beta_t, \alpha_t]^T$, where (e_t, n_t, u_t) is the position in the global coordinate frame and $(\gamma_t, \beta_t, \alpha_t)$ are the yaw, pitch and roll angles, respectively, of the vehicle in the same coordinate frame.

3 Models

For a filtering solution to the problem defined above, we need both a process model, describing how the state evolves over time, and measurement models that describe the relation between the state and our observations.

3.1 Process model

We model the vehicle using a simple point mass model. As we deem that the measurements from the gyroscope and wheel speed sensors are sufficiently accurate for our purposes we have chosen to include them directly in our process model. This model can be expressed in a difference equation,

$$\mathbf{P}(\mathbf{x}_t) = \mathbf{\Delta}_t \mathbf{P}(\mathbf{x}_{t-1}) \tag{1}$$

$$\mathbf{\Delta} = \begin{bmatrix} e^{[\Delta t \boldsymbol{\omega}_t + \mathbf{q}_t^\omega]_\times} & \Delta t \mathbf{v}_t + \mathbf{q}_t^v \\ \mathbf{0} & 1 \end{bmatrix} \tag{2}$$

where $\mathbf{P}(\cdot) \in SE(3)$ is the 4x4 matrix representation of a pose, $[\mathbf{a}]_\times$ is the 3×3 matrix such that $[\mathbf{a}]_\times \mathbf{b} = \mathbf{a} \times \mathbf{b}$ for all $\mathbf{b}$, Δt is the time between the samples enumerated by $t-1$ and t, and $\mathbf{q}_t = [\mathbf{q}_t^\omega, \mathbf{q}_t^v]^T = \mathcal{N}(\mathbf{q}_t; \mathbf{0}, \Delta t \mathbf{Q})$.

3.2 Measurement model

We will present a measurement likelihood given the set of feature points in the current image $\mathbf{f}_t$, for both types of maps used in this paper, semantic and SIFT.

To arrive at a concise description of the likelihood we here assume that we know the correspondence between the points in the map and the points in the current image. As such, we have a data association vector $\boldsymbol{\lambda}_t = [\lambda_t^1, \ldots, \lambda_t^{n_t}]^T$, where $\lambda_t^i = j$ indicates that image feature i corresponds to map feature j if $j > 0$, otherwise the feature is not present in the map. Using

this data association and assuming conditional independence between the pairs of $\mathbf{u}_t^i$ and $\mathbf{d}_t^i$, we get an expression for the likelihood as

$$\begin{aligned} p(\mathbf{f}_t|\boldsymbol{\lambda}_t, \mathbf{x}_t, \mathcal{M}) &= p(\{\langle \mathbf{u}_t^i, \mathbf{d}_t^i \rangle\}_{i=1}^{n_t}|\boldsymbol{\lambda}_t, \mathbf{x}_t, \mathcal{M}) \\ &= \prod_i p(\mathbf{u}_t^i, \mathbf{d}_t^i|\boldsymbol{\lambda}_t, \mathbf{x}_t, \mathcal{M}) \\ &= \prod_i p(\mathbf{u}_t^i, \mathbf{d}_t^i|\mathbf{x}_t, \mathcal{M}_{\lambda_t^i}). \end{aligned} \tag{3}$$

where $\mathcal{M}_{\lambda_t^i}$ denotes the 3-D point with associated descriptor and visibility parameters in the map $\mathcal{M}$ which corresponds to feature i according to the data association $\boldsymbol{\lambda}_t$. So to be able to express (3), we need to define the model $p(\mathbf{u}_t^i, \mathbf{d}_t^i|\mathbf{x}_t, \mathcal{M}_{\lambda_t^i})$ for our two types of maps. How we handle the fact that $\boldsymbol{\lambda}_t$ is not available from the measurements, is given in Section 4.

SIFT map

We start with a brief description of traditional localization in a point map with SIFT descriptors. With a given data association, the descriptor part of the feature will not contribute to the likelihood. We assume that the location of the SIFT detection in the image is subject to some noise, and a popular model for this is that the projection error of the 3-D points is zero mean and normally distributed,

$$\begin{aligned} p(\mathbf{u}_t^i, \mathbf{d}_t^i|\mathbf{x}_t, \mathcal{M}_{\lambda_t^i}) &\propto p(\mathbf{u}_t^i|\mathbf{x}_t, \mathbf{U}_{\lambda_t^i}) \\ &= \mathcal{N}(\mathbf{u}_t^i; \boldsymbol{\pi}(\mathbf{x}_t, \mathbf{U}_{\lambda_t^i}), \sigma_\pi^2), \end{aligned} \tag{4}$$

where $\boldsymbol{\pi}(\cdot)$ is a standard pinhole camera projection model with lens distortion[12], and σ_π^2 is the variance of the detector error. Both the mounting of the camera relative the vehicle coordinate frame and its intrinsic parameters are implicit in the $\boldsymbol{\pi}(\cdot)$ function.

Semantic map

In the case of the semantic maps, both the descriptor of each map point, D_j, and image feature descriptor d_t^i is a scalar class label from the Cityscapes classes [9], i.e., $D_j \in \{\text{Building}, \text{Road}, \dots\}$. Further, as the semantic segmentation gives a class label for each pixel in the image, $\mathbf{f}_t$ is dense in the sense that it contains all the pixels in the image.

Even though the descriptor for nearby pixels in the image clearly are correlated from the neural net classifier, we again make the simplifying

assumption that the pixel class and pixel coordinates are independent and can thus partition the likelihood for a single feature point as

$$p(\mathbf{u}_t^i, d_t^i | \mathbf{x}_t, \mathcal{M}_{\lambda_t^i}) = p(\mathbf{u}_t^i | \mathbf{x}_t, \mathbf{U}_{\lambda_t^i}) \Pr\{d_t^i | \mathbf{x}_t, \mathcal{M}_{\lambda_t^i}\}. \quad (5)$$

This factorization would lead to overconfidence in the observations, if not adjusted for, and this will be addressed in Section 4.2. The first factor, $p(\mathbf{u}_t^i | \mathbf{x}_t, \mathbf{U}_{\lambda_t^i})$, denotes the probability of detecting a feature in pixel i. However, since all pixels in the input image are classified by the segmenter, and all pixels are used in the semantic model, pixel i will always detect a feature and hence $p(\mathbf{u}_t^i | \mathbf{x}_t, \mathbf{U}_{\lambda_t^i})$ is constant for all i. We thus obtain

$$p(\mathbf{u}_t^i, d_t^i | \mathbf{x}_t, \mathcal{M}_{\lambda_t^i}) \propto \Pr\{d_t^i | \mathbf{x}_t, \mathcal{M}_{\lambda_t^i}\}. \quad (6)$$

Turning to the expression in the right hand side of (6), we have two cases: either there is no map point projected to this pixel, $\boldsymbol{\lambda}_t^i = 0$, or there is one, $\boldsymbol{\lambda}_t^i > 0$. In the first case, we have no information from the map about its class, and we assume a distribution for all such pixels which is simply the marginal distribution over all classes,

$$\Pr\{d_t^i | \lambda_t^i = 0, \mathbf{x}_t, \mathcal{M}_{\lambda_t^i}\} = \Pr\{d_t^i\}. \quad (7)$$

In the second case, the pixel coordinates corresponds to a point in the map but we are still uncertain if we detect the point or if it is occluded by something, e.g., a vehicle or pedestrian. To handle this uncertainty we introduce a detection variable δ which is 1 if the map point is detected in the image and 0 otherwise. Using this detection variable we can express the likelihood for the pixels with corresponding map points as

$$\Pr\{d_t^i | \mathbf{x}_t, \mathcal{M}_{\lambda_t^i}\} = \sum_{\delta \in \{0,1\}} \Pr\{d_t^i | \delta, D_\lambda^i\} \Pr\{\delta | \mathbf{x}_t, \mathcal{M}_{\lambda_t^i}\}, \quad (8)$$

where $\Pr d_t^i | \delta = 0, D_{\lambda_k^i}$ and $\Pr d_t^i | \delta = 1, D_{\lambda_k^i}$ are design PMF:s for the pixel class probability given that specific map point is occluded or visible, respectively. These are sensor specific models that also depend on the properties of the semantic segmentation algorithm used. The remaining model describes the probability that a given map point is visible and can be structured as

$$\Pr\{\delta = 1 | \mathbf{x}_t, \mathbf{U}_{\lambda_k^i}, \mathcal{V}_{\lambda_k^i}\} = v(\mathbf{x}_t, \mathbf{U}_{\lambda_k^i}, \mathcal{V}_{\lambda_k^i}) \rho_{\lambda_k^i} (1 - P_o), \quad (9)$$

where $v(\cdot)$ is a function that is one if $\mathbf{x}_t$ is in the visibility wedge of the map point and P_o is a design parameter specifying the probability that a visible

map point is occluded. The probability for $\delta = 0$ is found as the reciprocal of (9).

To conclude, the likelihood is factored into one part for the pixel coordinates $\mathbf{u}_k^i$ and one for the descriptor d_k^i. The first of the two factors is 1 for all pixels, and the second factor contributes to the product over all features in (3) in two ways depending on whether or not there is a map point projected in the i:th pixel, according to

$$\begin{aligned} p(\mathbf{f}_t|\boldsymbol{\lambda}_t, \mathbf{x}_t, \mathcal{M}) &= \prod_i p(\mathbf{u}_t^i, \mathbf{d}_t^i|\mathbf{x}_t, \mathcal{M}_{\lambda_t^i}) \\ &= \prod_{i\in\{i:\lambda_t^i>0\}} p(d_t^i|\mathbf{x}_t, \mathcal{M}_{\lambda_t^i}) \prod_{i\in\{i:\lambda_t^i=0\}} \Pr\{d_t^i\}. \end{aligned} \tag{10}$$

4 Algorithmic details

Now we have tractable models for both the motion and the two different classes of measurements based on SIFT descriptors and the semantic class descriptor. The measurement models are conditioned on a specific data association, and in the semantic class case, the model provides a simple way to make the correct data association, but for SIFT descriptor case we will describe the process further.

4.1 SIFT filter

Algorithm 1 SIFT based localization

1: initialize $\mathbf{x}_0$
2: **for** each time instance t **do**
3: acquire image $\mathbf{y}_t$
4: motion update (1)
5: extract SIFT points $(\mathbf{u}_t, \mathbf{d}_t)$ from $\mathbf{y}_t$
6: select local map $\mathcal{M}_t$ from $\mathcal{M}$ using $\mathcal{V}$ and $\hat{\mathbf{x}}_t$
7: match nearest neighbor $\mathbf{d}_t$ to $\mathcal{M}_t$
8: RANSAC on $\mathbf{u}_t$ and $\mathbf{U}$ from $\mathcal{M}_t$ to find $\boldsymbol{\lambda}_t$
9: measurement update (4)

Our reference localization algorithm is similar to [7], but instead of an iterative optimization we have implemented a UKF filter, and instead of iterative reweighting, we use RANSAC to select inliers from the proposal correspondences. The UKF filter makes use of the process model (1) and measurement model for SIFT features (4), and is described in pseudo code in Algorithm 1.

Before the SIFT detections can be used in (4), we must know the data associations, $\boldsymbol{\lambda}_t$. Recall that $\boldsymbol{\lambda}_t$ represents how point features are matched between the observed image and the map $\mathcal{M}$. We select one $\boldsymbol{\lambda}_t$ by first selecting a subset of possibly visible points, $\mathcal{M}_t$, from the full map, $\mathcal{M}$, based on the current pose estimate and the visibility of the map points $\mathcal{V}$. Then the observed SIFT descriptors are matched in the image to their nearest neighbors (L_2-distance in the descriptor space) in this local map, $\mathcal{M}_t$, using Lowe's ratio criterion [13] to select candidate matches. Then, to further cull false correspondences we employ a 3-point RANSAC approach in which three correspondence pairs are selected, and the four camera views that are consistent with these correspondences are calculated. Finally, we select the configuration giving the most inliers according to the reprojection error being less than a certain threshold, in our case 6 pixels.

4.2 Semantic filter

Algorithm 2 Sematic class based localization

initialize particles $\mathbf{x}_0$ and weights $\mathbf{w}_0$
for each time instance t **do**
  acquire image $\mathbf{y}_t$
  motion update (1)
  assign class $\mathbf{d}_t^i$ to each pixel of $\mathbf{y}_t$
  select local map $\mathcal{M}_t$ from $\mathcal{M}$ using $\mathcal{V}$ and $\mathbf{x}_t$
  measurement update (12)
  normalize weights $\mathbf{w}_t$, and resample if needed

For the localization using semantic data, we have chosen a bootstrap particle filter [14] to recursively estimate the posterior distribution as a sum of weighted Dirac delta functions. A particle filter is in this case more suitable than, for example, a UKF as the likelihood is a probability mass function.

To be able to evaluate the likelihood for a particle, we first need to determine which points in the map are potentially visible. This is similar to what is done for the SIFT case, and only needs to be done approximately and can thus be calculated for several nearby particles simultaneously, e.g. using their mean position together with the visibility parameters, $\mathcal{V}$, from the map. The potentially visible points are then projected to the image plane, creating a unique assignment, $\boldsymbol{\lambda}_t$, from map to pixels for each particle. An illustration of map points projected into the segmented image is provided in Fig. 3.

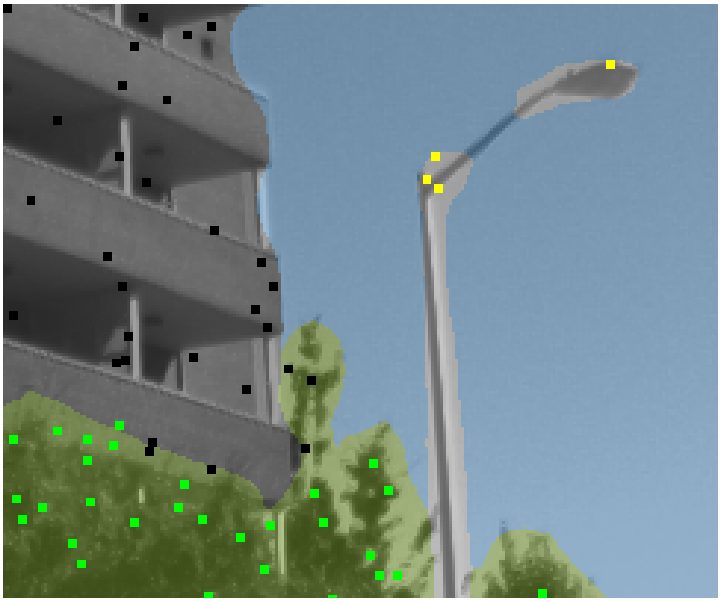


Figure 3: Cropped area from a segmented image with map points projected into it using the mean pose provided by the semantic localization filter. Mapping and localization are in this case separated in time by only 2 weeks. Black dots are map points which are classified mainly as building, green represents vegetation, and yellow is from the pole class.

Dividing (10) by the constant $\prod_i \Pr d_t^i$ will simplify the weight update since we then only have to consider the pixels with a point projected into them,

$$p(\mathbf{f}_t|\boldsymbol{\lambda}_t, \mathbf{x}_t, \mathcal{M}) \propto \frac{\prod_{i\in\{i:\lambda_t^i>0\}} p(d_t^i|\mathbf{x}_t, \mathcal{M}_{\lambda_t^i})}{\prod_{i\in\{i:\lambda_t^i>0\}} \Pr\{d_t^i\}}. \tag{11}$$

Because we chose to model the measurements as conditionally independent when they are in fact not, the update we would get from this would be overly confident in the measurement, and to reduce this effect we raise the measurement likelihood to a positive number smaller than 1, so that the weight update for particle j becomes

$$w_t^{(j)} \leftarrow w_{t-1}^{(j)} \times p(\mathbf{f}_t|\boldsymbol{\lambda}_t, \mathbf{x}_t^{(j)}, \mathcal{M})^{s/\max\{n_{\lambda_t}, N_c\}} \tag{12}$$

where $w_t^{(j)}$ is the weight associated with the j:th particle with state $\mathbf{x}_t^{(j)}$, s is a scaling parameter, n_{λ_t} is the number of map points that are projected in the image, and N_c is a cutoff where more projected map points in the image do not contribute with more information, with the rationale that more points means their spacing in the image is smaller and thus their corresponding measurements are more correlated to each other.

The top level algorithm is summarized in pseudo code in Algorithm 2.

5 Evaluation

The localization framework is evaluated on the Carnegie Mellon University (CMU) visual localization dataset [10, 15], where a test vehicle equipped with two cameras traversed a route of approximately nine kilometers in Pittsburgh sixteen times throughout the period September 1, 2010 - September 2, 2011. The route consists of a mix of urban and suburban areas, as well as green parks where mostly vegetation is visible in the cameras. We have used 12 of these 16 sequences in our evaluation. The selected runs capture the changes of the environment throughout the seasons, as well as a variety of weather and lighting conditions. The SIFT-features were extracted using VLFeat [16], and the semantic segmentation was done using Dilation 10 [17].

5.1 Map creation

For camera localization to work, we need a map to localize in. The focus of this paper is on the localization models, but we will give a small note on how we have chosen to handle the map, since there is no map included in the dataset. We have picked the first sequence of measurements from 1 September 2010 and created a map from that sequence of images. The remaining sequences are not used in the map creation, but only used to evaluate localization with respect to this map from Sep. 1. We used a structure-from-motion pipeline based on [18], with the GPS and odometry constraints used to create an initial trajectory, after which a bundle adjustment procedure gave the final solution to landmarks and poses. Images at standstill and very low speeds were culled in order to avoid unnecessary computations. Because of limited computer resources, the whole sequence was split into 38 smaller parts that were mapped separately.

After calculating the 3-D points and camera poses, the descriptors for each point are determined. For the SIFT map, the arithmetic mean of the descriptors corresponding to each view of the 3-D point is taken as the descriptor of the 3-D point. For the semantic map, a small neighborhood of 7×7 pixels around the detected point in each image is taken and then a normalized histogram over the classes of those pixels is used as the PMF $\Pr\{d_t^i \big| \delta = 1, D_{\lambda_k^i}\}$ directly. The marginal PMF from (7) is also calculated from data, as the normalized histogram for all pixels in all images in the mapping sequence. The last design PMF, $\Pr\{d_t^i \big| \delta = 1, D_{\lambda_k^i}\}$, is a manual adjustment of the marginal PMF. The dynamic objects, such as cars and

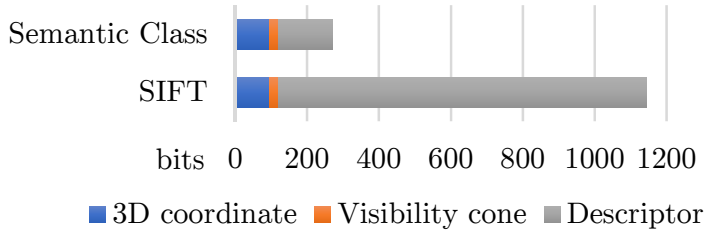


Figure 4: Storage space required for each point in the map. 3-D point and visibility cone are needed regardless of descriptor. The descriptors are quantized to 8 bit resolution.

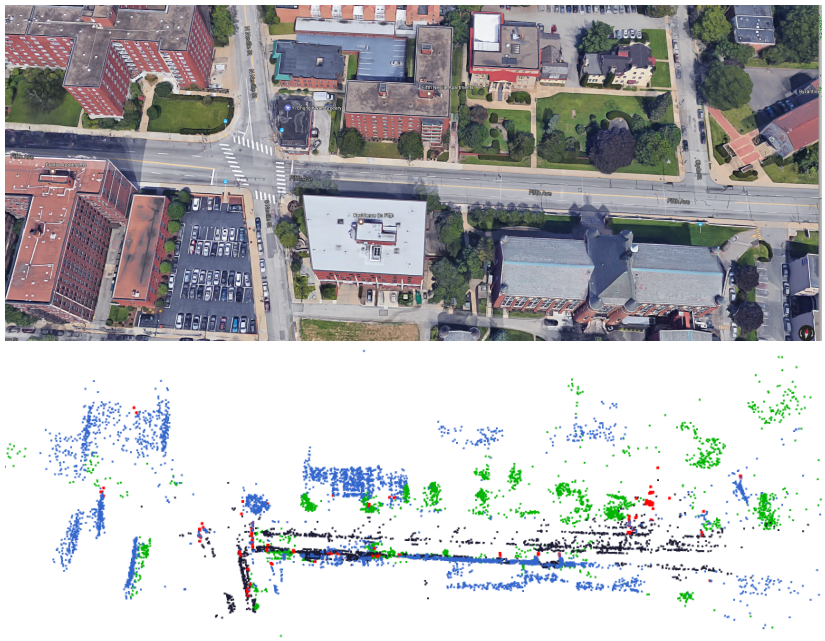


Figure 5: One part of the map, as viewed in Google Maps 3-D view (top), and the point cloud result of the structure from motion solution (bottom) colored by most likely category where blue is "construction", green is "nature", black is "flat" and red is "stationary object".

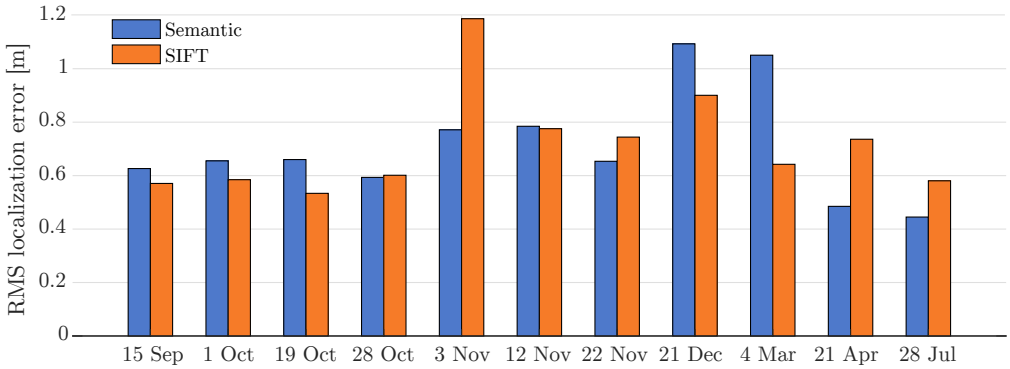


Figure 6: Root mean square error of the eleven runs for both the semantic class based localization and SIFT based localization.

pedestrians get increased probability while the stationary objects get decreased. We can see in fig. 4 that the size of SIFT descriptors is more than 6 times larger than even the most naive way of storing the semantic class descriptor. We have observed that normally each point only has probability mass in three or fewer classes, so if we encode only the top three most probable classes for each point, the descriptor size can be reduced to 39 bits. In fig. 5, we show an example of the resulting point cloud from the map creation with semantic categories indicated by color.

5.2 Ground truth

The dataset provides GPS measurements and some form of ground truth data which includes pose, but the ground truth trajectories were not completely accurate. In order to obtain a more reliable ground truth, the sequences were aligned by the same structure from motion pipeline as in the map creation using the SIFT descriptors already available in each map. Typically, matches between sequences are much less frequent than within a sequence, and in 147 out of a total of 418 (11×38) pieces, not enough matches to the map from Sep. 1 were found to produce a ground truth with confidence. These sequences were then excluded from the evaluation. The excluded portions seemed contain much vegetation, where SIFT is known to not be very reliable, and thus we would expect the reference algorithm to also perform badly in these areas compared to where we were able to construct a reliable ground truth.

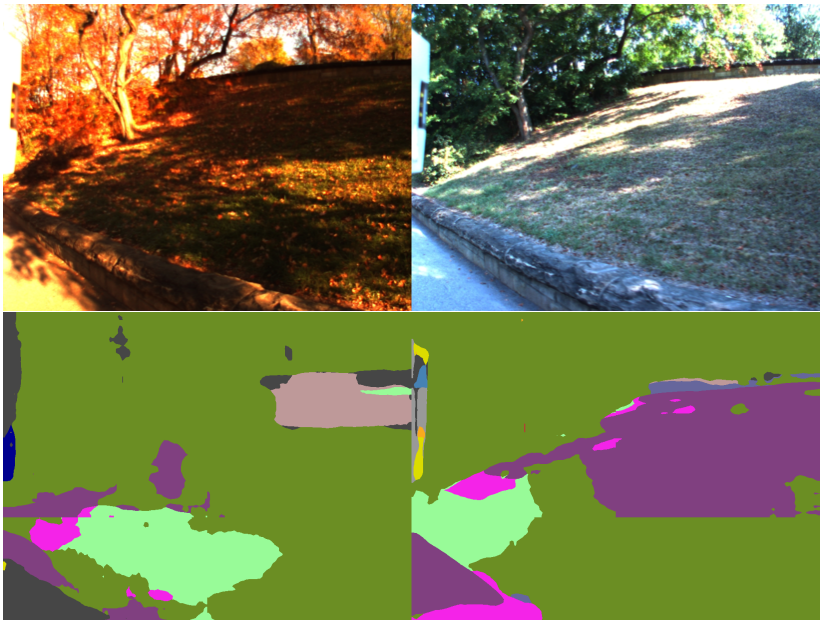


Figure 7: Image and its segmentation for the place where semantic localization fails. Localization image to the left and mapping image to the right.

6 Results

Overall semantic localization performed on par with the reference in our evaluation. In one of the 271 parts with available ground truth, the semantic localization algorithm suffered a failure from which it was not able to recover. For the results below, only the non failing localization runs are included. Fig. 6 shows the root mean square (RMS) error in meters for the two approaches for the different test runs.

When looking into what causes bad localization for the SIFT based algorithm, we notice faulty data association and extended time periods of not enough inliers after the RANSAC as the main causes of failure. When looking into the failure cases for the semantic localization we see mainly large areas misclassified by the segmentation algorithm, or only very few classes covering most of the image in such a way that the pose is not sufficiently constrained in all dimensions, see example in Fig. 7.

7 Discussion

The results are promising in the sense that we can perform localization with results comparable to the reference algorithm, despite using much less informative point descriptors in the map. The results seem to contradict

the goal of increased robustness, but from the typical failure cases we have observed we believe that with improved segmentation algorithms trained on data obtained during a larger range of environmental conditions (for example during winter, in more extreme lighting conditions and so on), and also possibly including more classes by, e.g., adding road markings, splitting the vegetation into trunk/large branch and foliage, we would see an improvement in failure cases of the semantic localization.

Bibliography

[1] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: A survey", *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, 2015.

[2] C. Valgren and A. J. Lilienthal, "SIFT, SURF & seasons: Appearance-based long-term localization in outdoor environments", *Robotics and Autonomous Systems*, vol. 58, no. 2, pp. 149–156, 2010.

[3] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Second. Cambridge University Press, ISBN: 0521540518, 2004.

[4] Y. Li *et al.*, "Worldwide pose estimation using 3D point clouds", in *European Conference on Computer Vision*, 2012.

[5] T. Sattler, B. Leibe, and L. Kobbelt, "Improving Image-Based Localization by Active Correspondence Search", in *ECCV*, 2012.

[6] H. Aanæs, A. L. Dahl, and K. Steenstrup Pedersen, "Interesting interest points", *International Journal of Computer Vision*, vol. 97, no. 1, pp. 18–35, Mar. 2012.

[7] P. Mühlfellner *et al.*, "Summary maps for lifelong visual localization", *Journal of Field Robotics*, vol. 33, no. 5, pp. 561–590, 2016.

[8] K. M. Yi *et al.*, "Lift: Learned invariant feature transform", in *European Conference on Computer Vision*, Springer, 2016, pp. 467–483.

[9] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding", in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[10] H. Badino, D. Huber, and T. Kanade, "Visual topometric localization", in *Intelligent Vehicles Symposium (IV)*, Baden-Baden, Germany, Jun. 2011.

[11] G. Pascoe, W. Maddern, and P. Newman, "Direct visual localisation and calibration for road vehicles in changing city environments", in *IEEE International Conference on Computer Vision: Workshop on Computer Vision for Road Scene Understanding and Autonomous Driving*, Santiago, Chile, Dec. 2015.

[12] D. C. Brown, "Close-range camera calibration", *Photogramm. Eng*, vol. 37, no. 8, pp. 855–866, 1971.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[14] B. Ristic, S. Arulampalam, and N. J. Gordon, *Beyond the Kalman filter: Particle filters for tracking applications*. Artech house, 2004.

[15] H. Badino *et al.*, "Real-time topometric localization", in *International Conference on Robotics and Automation (ICRA)*, St Paul, Minnesota, USA, May 2012.

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

[17] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions", in *ICLR*, 2016.

[18] O. Enqvist, C. Olsson, and F. Kahl, "Non-sequential structure from motion", in *Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras (OMNIVIS)*, 2011.

Paper IV

Long-term 3D Localization and Pose from Semantic Labellings

C. Toft, C. Olsson and F. Kahl

3D Reconstruction Meets Semantics (3DRMS) Workshop at the International Conference on Computer Vision (ICCV) 2017.

Comment: The layout of this paper has been reformatted in order to comply with the rest of the thesis.

Long-term 3D Localization and Pose from Semantic Labellings

C. Toft, C. Olsson and F. Kahl

Abstract

One of the major challenges in camera pose estimation and 3D localization is identifying features that are approximately invariant across seasons and in different weather and lighting conditions. In this paper, we present a method for performing accurate and robust six degrees-of-freedom camera pose estimation based only on the pixelwise semantic labelling of a single query image. Localization is performed using a sparse 3D model consisting of semantically labelled points and curves, and an error function based on how well these project onto corresponding curves in the query image is developed. The method is evaluated on the recently released Oxford Robotcar dataset, showing that by minimizing this error function, the pose can be recovered with decimeter accuracy in many cases.

1 Introduction

In 1982 Marr's unified theory of vision [1] was published and it has been a major source of inspiration to the vision community. The theory resembles human perception and works on multiple levels; starting with local visual primitives and ending with a global understanding of the scene. Interestingly, when examining today's best performing visual mapping [2–4] and localization [5, 6] systems, the overall understanding of the scene is largely lacking. Instead they rely on the geometry of point projections and the availability of local features that are descriptive enough to be uniquely and reliably matched across images, without any semantic understanding.

The reliance on local texture descriptors makes the system sensitive to viewpoint changes, weather conditions, lighting and seasonal variations etc. that all affect local scene appearance. Additionally, without any high level understanding it is hard to determine which parts of the scene may be unreliable for localization such as cars or other moving objects. As a consequence traditional geometric localization systems are insufficiently constrained under weak local appearance information.

Figure 1: Two examples of successfully localized pictures. In the left column, the query images are shown together with the reprojection of the 3D curves corresponding to road edges and poles. The images on the right show approximately the same location as seen in the mapping sequences. Note that our baseline method based on LIFT features failed to obtain consistent 2D-3D matches.

This paper addresses the fundamental question "Is it possible to perform image localization from high level information, such as a semantic understanding of the image content?" Such information is in contrast to local texture largely invariant to weather, lighting and seasonal changes. We leverage the recent progress in pixelwise semantic image labelling to obtain robust scene descriptions suitable for long-term localization. The basic idea is that the distribution of semantic classes in the query image should alone be sufficient to provide strong constraints on the camera pose.

To solve the problem we create a scene model consisting of simple geometric primitives, such as 3D points and curves, but with a meaningful semantic label. These are projected into the query image and compared to its semantic content. Our results show that this simple approach can be used for reliable long-term localization from a single query image, see Figure 1 for two examples. As we are only using semantically labelled information in the query image, the added invariance allows us to localize images captured under completely different conditions than the model. One may argue that we are not using all the information present in the query image, as we only rely on the semantic labels. This is of course correct, and in a practical system one should use all the available information in the query. In this work, we are pushing the limits and investigating if it is possible to achieve reliable camera pose estimation at all under these conditions. Our

experimental results on long-term 3D localization in urban street scenes are quite encouraging. We show that one can in many cases achieve global, metric localization from a single image despite variations in seasons and challenging lighting conditions where localization approaches based on local features fail completely.

2 Related work

Traditionally, camera pose estimation (sometimes called "camera resectioning", or simply "localization") is performed by matching point features between the query image and the 3D model. In this case, the model simply consists of a set of points in three-dimensional space. Associated to each point is one or more descriptor vectors, describing the local appearance of the point as it was seen when the 3D model was constructed. When a picture is to be localized, feature points are extracted from the image, and each image point is matched to the most similar point in the 3D model. In this way, a set of 2D-3D correspondences are obtained, from which the full six degrees-of-freedom pose can be calculated [7]. This is in contrast to approaches working in the image domain only, solving the problem of "visual place recognition", see [8] for a survey. We will only be concerned with the 3D localization problem.

The main problem that makes long-term localization difficult is the fact that the feature descriptors used to describe the image and the 3D scene are not invariant to the changes in environment seen during different seasons, weather and time of day (such as SIFT [9], ORB [10] and SURF [11]). Valgren and Lilienthal [12] examined the suitability of SIFT and SURF for long-term localization from a single image and found that the upright U-SURF performed best for their scenario. Another way to approach the long-term localization problem is to find a new descriptor that better copes with changes of the environment. For example, Yi *et al* [13] created a new feature descriptor, called LIFT, by training a convolutional neural network on image patches corresponding to the same feature but viewed under different ambient conditions, and found that this descriptor generated more correct matches between pictures taken under very different lighting conditions compared to SIFT. We use the LIFT descriptor for baseline comparisons to our approach as LIFT outperforms many competing feature descriptors by a large margin including SIFT.

Badino *et al* [14, 15] performed cross-seasonal visual localization on a nine kilometer stretch of road in Pittsburgh. The road was traversed more than a dozen times throughout the span of a year, capturing seasonal variations and a variety of weather conditions. A map was created using one of

the traversals, storing the GPS location and the SURF features visible in the camera at more or less equally spaced locations on the road. Localization could then be performed on the remainder of the datasets using a Bayesian filtering approach. The approach is hence dependent on the invariance of the SURF descriptor. To compensate for feature matches being unreliable, a sequence of consecutive images was used to perform localization. The idea of using multiple images for localization (or rather place recognition) was also pursued in [16] where up to 300 consecutive images were used to perform localization based on a image intensity correlation measure. Self-localization using only visual odometry information was investigated in [17].

There are a number of elaborate 3D localization algorithms from a single image that have been developed for handling large rates of incorrect matches, see [5, 6, 18–25]. Still, if local feature matching is not working properly, such approaches are doomed to fail. In [26], a mining approach is applied to find stable local features over time. Deep learning approaches are presented in [27, 28]. In [29] an information-theoretic metric is derived to compare the query image and a rendered image without relying on individual pixels for the purpose of long-term visual localization. Though it requires a complete geometrical 3D model of the environment. We explore an alternative route to obtain cues that are reliable in the long run by using semantic information.

In [30], object recognition in indoor scenes is applied to obtain more stable matches for robot localization in a 2D map. The approach is based on particle filtering, which means that multiple observations over time are needed. Another source of inspiration for our work is on semantic 3D reconstruction [31]. Here it is shown that 3D reconstruction and multi-view stereo can be supported by using semantic labellings in the image.

3 A motivating example

The input to the localization algorithm is the pixelwise semantic labelling of the query image. If the camera pose can be computed accurately using only this information, then 3D localization can be performed robustly under varying environmental conditions, provided that the method used for semantic segmentation outputs accurate labels under these conditions. The localization problem is thus moved over to the segmentation itself, making accurate long-term localization a natural consequence of the progress in semantic labelling.

Figure 2 shows a typical semantic labelling of an image from the Oxford Robotcar dataset [32], where the labelling has been obtained by applying the method described in [33] trained on the Cityscapes dataset [34]. The

Figure 2: A typical example of a pixelwise semantic labelling of a picture from the Oxford Robotcar dataset.

labelling consists of a single integer for each pixel, denoting the semantic class assigned to it. The classes commonly include road, pavement, buildings, vegetation, poles and sky, among others. Note also that the different connected components in the image are completely featureless and fully characterized by their contours.

Through inspection of the image, one might expect to be able to extract two kinds of pose information from the image. The course spatial distribution of semantic classes in the image should be able to provide rough information about where in the map the image is taken; pictures taken in parks would be dominated by vegetation, whereas pictures taken in the city center would likely contain considerably more buildings.

However, it also seems reasonable to expect to be able to extract more precise metric information as well. The road and the contour where the sky meets the distant vegetation provide information about the camera rotation, the two edges of the road provide information about the lateral position of the car on the road, and the poles on the side of the road should provide accurate information about the longitudinal position along the road. Taking all the evidence into account, it should thus be possible to calculate the full six degrees-of-freedom pose from a single labelled image. In the following section, we present a framework that handles this information in a unified manner and allows efficient pose calculation by minimization of a loss function.

4 Framework for semantic localization

4.1 Model

Our model consists of two types of primitives; 3D points and space curves. The 3D points $\{X_i\}_{i=1}^{M}$ are each assumed to belong to a single semantic category and therefore have an associated label. Given a candidate 3×4 camera matrix P (which encodes both orientation and position) we compute the projection PX_i and penalize $d_{L_i}(PX_i)$, where $d_{L_i}(x)$ measures the distance between x and the closest pixel in the image labelled L_i. Note that for a pixel labelled L_i the absence of a correctly labelled projection does not incur any penalty. It is only when a 3D point is projected into a semantically different segment that a penalty occurs. This is essential since our 3D models are built using standard SfM systems and are therefore far from complete. Additionally, this allows us to handle occlusion in a very simple but effective way by recording at what distances a 3D point should be seen and adding a depth threshold to the $d_{L_i}(PX_i)$ term.

Since much of the information in a semantically labelled image is stored in the curves separating different classes, our 3D model also includes a set of space curves $\{\mathcal{C}_i\}_{i=1}^{N}$ endowed with two semantic labels L_i^1 and L_i^2. For the 3D curves we use a penalty $\int_{P\mathcal{C}_i} \eta_{L_i^1,L_i^2}(x(s))ds$, where $\eta_{L_i^1,L_i^2}(x)$ is a function that computes the smallest truncated distance between the point x and an image curve separating regions labelled L_i^1 and L_i^2. Note that our space curves may not correspond to actual physical curves. While the curve separating road and sidewalk is real the skyline is not. We still found that using these and treating them as curves far away helps to constrain the localization. In particular they are useful for determining orientation.

Similar to the 3D points the curves in the 3D model do not need to explain the entire observed image. For example, if we wish to use the skyline where the distant vegetation meets the sky as a curve type (as we do in the experimental section), we are not penalized if the skyline curve in the 3D model is not reprojected onto the entire observed skyline in the image. Instead, we are only penalized for every point where the projection of the 3D curve representing the skyline does not coincide with the observed skyline in the query image.

Our complete loss function is of the following form:

$$E(P) = \sum_{i=1}^{N} \lambda_{L_i^1,L_i^2} \frac{1}{l_i} \int_{P\mathcal{C}_i} \eta_{L_i^1,L_i^2}(x(s))\mathrm{d}s \; + \tag{1}$$
$$+ \sum_{i=1}^{M} \gamma_{L_i} \frac{1}{M_{L_i}} d_{L_i}(PX_i),$$

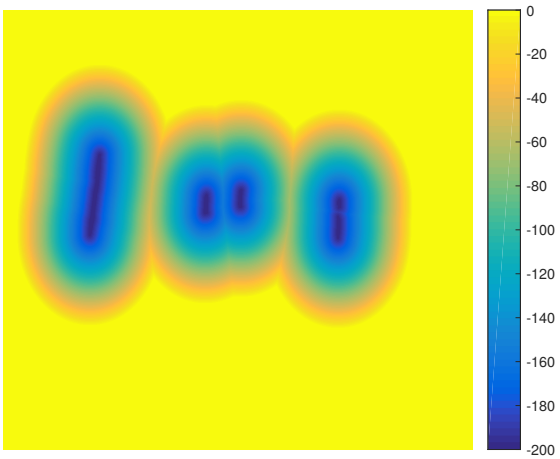


Figure 3: Error map η for the class "poles" in the image in Figure 2.

where the integral is computed with arc-length parametrization in s. The numbers $\lambda_{L_i^1,L_i^2}$ and γ_{L_i} are weights for the different semantic classes, giving us the choice to give some evidence more weight than other, if desired. In the experiments performed in Sec. 5, these constants were all set to one. l_i is the length of the reprojected curve i. The value M_{L_i} is the number of points seen in the image with label L_i. The loss function (1) can be evaluated very efficiently by storing the distance functions $\eta_{L_i^1,L_i^2}$ and d_{L_i} in a look-up table, as shown in Figure 3. When we wish to evaluate E for a given pose P, the curves and points are projected into P, and then the corresponding values for η and d are retrieved from the pre-computed table. This makes iterative minimization of (1) very fast.

In the framework presented above, we have not specified what types of curves to use for localization. In the localization experiments in Section 5, the curves $\mathcal{C}_i$ were piecewise linear curves, since these are very simple to project into the cameras and integrate over. We have also not specified any specific semantic labels for the curves yet. The only requirement for them to be useful is that it should be possible to reliably extract these curves from a semantically labelled picture.

4.2 Optimization of loss function

The loss function is a complicated, non-convex function with many local minima. In order to find a good minimum of (1), some prior knowledge about the problem structure must be utilized. Otherwise, if gradient descent is performed on an initially estimated camera pose, we run a high risk of ending up in a local minimum unless the initial pose happens to be very close to the global minimum. In the experiments presented in the next section,

we used curves representing the two edges of the road and poles along the street, as well as curves representing the contours of distant trees across the sky. Exploiting this knowledge, the following procedure was performed to minimize $E(P)$.

Given an initial estimate of the pose, gradient descent is performed on (1) using only points and the road edges, the terms for the other lines set to zero. This will likely yield good estimates for the camera rotation. This is followed by gradient descent where only the terms corresponding to road edges and poles are included.

At this stage, the rotation and lateral position of the car are likely close to their optimal values. It is thus reasonable to assume that five of the six degrees-of-freedom have been fixed: three for rotation, one for the lateral direction, and one for the vertical direction. Since only one degree of freedom remains, a line search is performed along this dimension, corresponding to the longitudinal position of the car on the road. This direction is assumed to be along the principal axis of the current camera. Figure 5 shows an example of the loss function along this direction. Finally, a last round of gradient descent is performed from the minimum obtained during the line search, keeping all terms of the loss function. All the derivatives for the gradient descent method are computed numerically.

The final question that must be addressed is where to obtain the initial estimate P_0 of the camera matrix. In a practical application, such as in a real autonomous driving scenario, there is probably a quite good estimate of the car position available from GPS (and other sensors) and internal odometry that could be used as a starting point for the local optimization. However, in this paper we perform global localization from a single labelled image with no other information, and use a simple initialization method based on the spatial statistics of the semantic labels in the query image.

Specifically, the top half of the segmented image is divided into six identically sized regions (two rows and three columns). Each region is then assigned a descriptor vector by making a histogram over all pixel classes in the region (excluding cars and pedestrians), and then normalizing the vector. To this vector, the two gradient histograms are then appended which are obtained from the binary images corresponding to the building and vegetation classes seen in the region, after being normalized and scaled by a factor 1/2. Finally, the six vectors obtained from all regions are stacked into a final descriptor vector.

During construction of the 3D map, this descriptor vector was calculated for all images in the mapping sequence. When later presented with an image to localize, the descriptor was calculated for the query image, and then matched to the closest descriptor from the mapping sequence. The found

Dataset	Date of collection	Purpose	Weather	Number of images
1	2014-05-06	Map building	Cloudy, diffuse lighting, few shadows	160
2	2014-05-06	Localization	Similar to above	188
3	2014-05-06	Localization	Mostly cloudy, but some sun and shadows	179
4	2014-11-28	Map building	Overcast, diffuse lighting, few shadows	46
5	2015-02-03	Localization	Winter, snow, some sun	71

Table 1: The five datasets used for evaluating the semantic localization algorithm. The top three datasets represent the same physical road, traversed three times during the same day, and the last two datasets represent a different road, traversed during two different seasons.

camera was then used as the initial camera matrix P_0 for local optimization.

5 Experiments

The presented framework for localization from semantically labelled images was evaluated on the Oxford Robotcar dataset [32]. Two different locations were used for the experiments. The first was a stretch of road approximately one hundred meters long and was traversed three times in slightly different weather conditions during May 6, 2014. The second sequence was approximately 70 m long and used to evaluate cross-seasonal localization. The data collected on November 28, 2014 was used to build the 3D map, and the data collected on the February 3, 2015 was used for 3D localization. Table 1 contains some more information about the individual datasets.

To generate a gold-standard localization reference, all the sequences were reconstructed using the publicly available structure from motion pipeline described in [35]. By manually adding 2D correspondences between pairs of sequences where necessary, all trajectories were reconstructed in the same coordinate system. Note that adding manual correspondences was a necessity as there were very few correspondences across the sequences. Bundle adjustment was then applied to all points and cameras simultaneously.

The first sequence of each location (i.e., datasets 1 and 4 in Table 1) was used as a reference - so called mapping sequence - from which semantic 3D maps were created, as will be explained below, and then the remaining sequences were used to evaluate the localization algorithm. Since no ground truth camera matrices are available in the dataset, the camera matrices obtained for the test sequences after bundle adjustment were used as a gold standard reference that the semantic localization could be compared against.

Piecewise linear three-dimensional curves of three different types were

Figure 4: An image from the mapping sequence, together with the reprojections of the 3D curves in the model.

reconstructed from the two mapping datasets. The different curve types used were road edges, poles and distant vegetation-sky intersections. Figure 4 shows an image from the mapping datasets, where the 3D model has been projected down into the camera. Note that the semantic 3D model is sparse in the sense that it contains few elements and does not cover all the imaged semantic content. As all space curves are piecewise linear, they are represented as a discrete set of points. The poles are thus represented by their start and end points, the road edges consist of around 100 3D points each.

The vegetation-sky curve might at first seem like a strange choice to include as a space curve, but it was found that the distant skyline was extracted from the semantic segmenter with remarkable consistency. Note also that if we can successfully match it to a curve in the observed image, we have fully determined the camera rotation. The only drawback compared with the other curves used is that the curve is not valid when the camera gets close to the curve. This turned out to not be a big problem in practice, since by the time it is no longer accurate, it has vanished from the top of the image and is no longer visible.

The road edge was automatically reconstructed by extracting four points on the road-pavement intersection in the 2D mapping image (using the semantic labellings), identifying the 3D points visible within the obtained quadrilateral, and then fitted a (road) plane through the corresponding 3D points using RANSAC [36]. The four corner points identified in the picture were then added to the 3D road curve. This procedure was repeated through the mapping sequences.

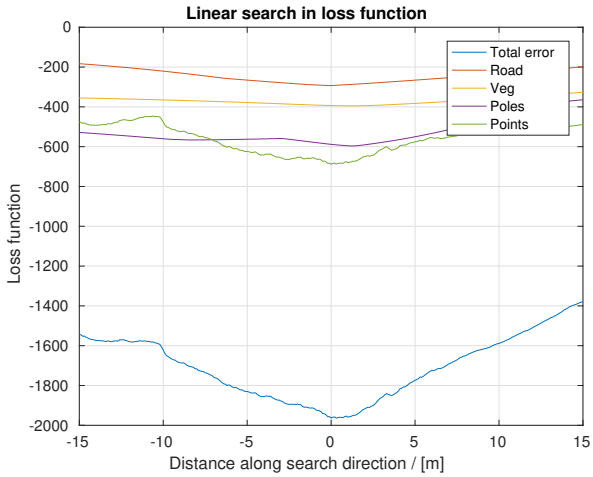


Figure 5: Example of cross-section of the loss function along the longitudinal direction. This line search is performed after the lateral direction and rotation have been established through gradient descent on the terms representing road edges in Eq. (1).

The poles were automatically reconstructed by tracking the corresponding connected components of the segmented pictures in the mapping sequence. Lines were then fitted to each pole in each image using a Hough transform. A line in 3D space was then obtained for each pole by back-projecting each observed 2D line and finding the intersection between these planes. The top and bottom points on the 3D lines were extracted based on at what height the top and bottom points of the pole were seen in the images. The 3D points of the model were obtained by triangulating consistent SIFT matches in the mapping sequences, where consistent here means that the 3D points satisfy the epipolar geometry and that they project to the same semantic label in all the visible mapping images.

The vegetation-sky curves were manually extracted by selecting a piecewise linear arc in an image. The 3D points seen in the image near that region were then retrieved, and the 3D curve was placed at a depth equal to the median depth of those points.

To perform localization of a single query image, the following procedure is followed. First, the image is semantically labelled. From this labelling, the error maps $\eta_{L_i^1,L_i^2}$ and d_{L_i} are calculated (cf. Figure 8). An initial estimate for the camera matrix P is then obtained, from which local optimization of (1) is performed. All constants λ and γ in (1) were set to one.

6 Results

The localization results are shown in Figure 6. Localization was performed on every image in the test datasets, each image being treated completely independently from the others. The results for datasets 2 and 3 (cf. Table 1) are shown together, and contain a total of 367 query images. The winter sequence (dataset 5) contains 71 images in total. The top left histogram over translational errors only show errors up to 10 m, but there were 20 outliers with translational errors greater than this, corresponding to bad initializations by the histogram matching procedure described at the end of Section 4. The rotational error histogram in the left column show all rotational errors, while in the winter road sequence (right histogram), there were 5 outliers outside the range shown. For the translational errors in the winter road sequence, there are only three outlier images outside the range shown in the translational error histogram.

The bottom row in Figure 6 shows a comparison with a three-point RANSAC using LIFT features. For a given value on the x-axis, the y-axis shows what fraction of the test images were localized to within the given value of x.

A few remarks are in order. First, for the first two test datasets, the localization accuracy is reasonably good. The translational error is less than a meter for around 73% of the images, and it is within two meters for 89%. The rotation was recovered within 2° degrees for 89% of the images.

When an image is successfully localized by LIFT, it is in general much more precisely localized than it is by the semantic localization method presented here. LIFT often recovers the pose with centimeter accuracy, whereas a pose constrained by several clear curves in our model tends to be localized within a few decimeters or half a meter. This is not very surprising, since the semantic features are more smeared out across the image than point features, and when looking at a given semantic segmentation, there often exists a rather large ambiguity as to where, for example, the poles and road edges actually are located.

For the winter road sequence, the localization errors for both LIFT and the current method were much larger than in the other test sets. In the first sequence, all three datasets were collected during the same day, so the dataset used to create the map was similar in appearance to the two test datasets. However, for the second sequence, the mapping dataset used to create the semantic 3D map was collected in late fall during a day when there was no snow, and the test dataset was collected in February during a snowy day, so the mapping and the test datasets appear very different.

The semantic localization algorithm mostly failed due to inaccurate seg-

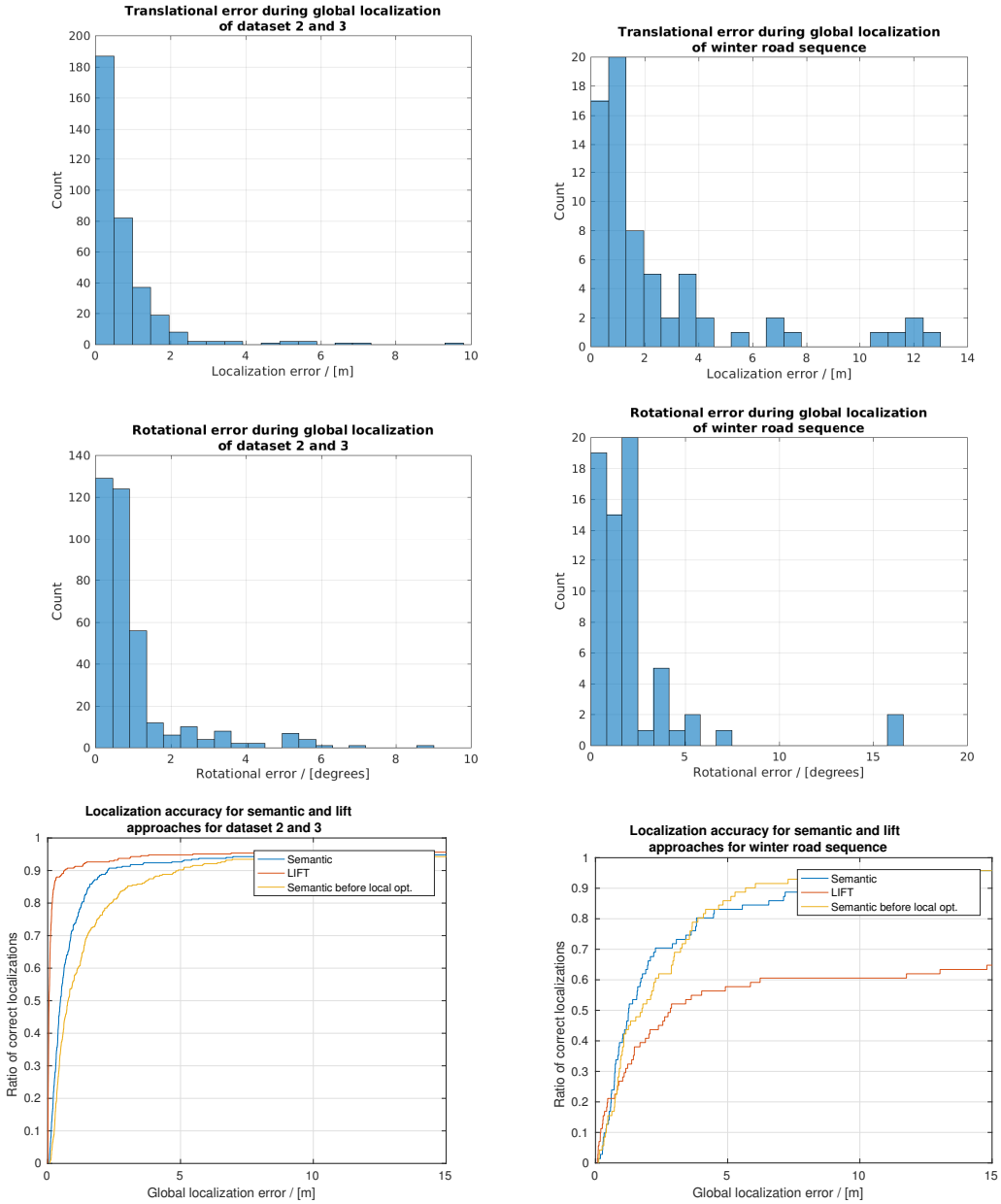


Figure 6: Localization results for the three datasets. The results for datasets 2 and 3 were similar, and have therefore been merged together. The first row shows histograms over the translational localization errors, and the second row shows the rotational errors. On the third row, a comparison is made with an approach based on LIFT point features. For a given value on the x-axis, the corresponding y-value gives the proportion of localizations with a translational error less than the x-value. For example, for datasets 2 and 3, 90% of the images were retrieved with 2.5 m or less translational error. Also shown is the translational errors before any local optimization is performed (i.e., using only the image retrieved by the semantic retrieval initialization method).

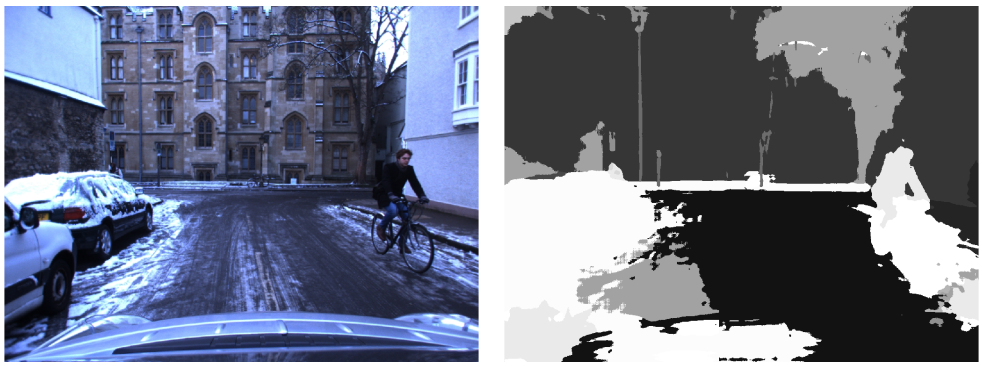


Figure 7: A failure case: No pavement was labelled as pavement, yielding no road edges that could be used for localization.

mentations. Most public datasets for street-view segmentation do not contain winter scenes, making the segmenters less accurate on these scenes. For example, the snowy ground was often misclassified, typically as a car, which made the loss function inaccurate since one of its terms evaluated how well the 3D curves corresponding to the road edges project down onto the road edges as seen in the query image. Figure 7 shows an example that our algorithm failed to localize. No correct road edges were detected, and the only pole that was correctly segmented was a drain pipe on the house in the background. Since very little useful information could be extracted from the input image, the localization failed.

Overall, we have seen that when the segmentation is accurate, it is generally possible to recover the camera rotation and translation with good accuracy, confirming that the semantic labelling conveys very strong information about the camera pose. See Figures 1 and 8 for examples of successful localizations.

7 Conclusion

We have considered the problem of how much pose information is stored in the semantic labels of a picture alone. We presented a method for performing full camera pose estimation based only on the pixelwise semantic labelling of the query image, and saw that in situations where the labelling is accurate, it is possible to recover the camera translation to within a few decimeters or meters accuracy, depending on the quality and location of the features observed, and the camera rotation to within a few degrees. We have thus shown that a good semantic segmentation provides very strong constraints on the camera pose.

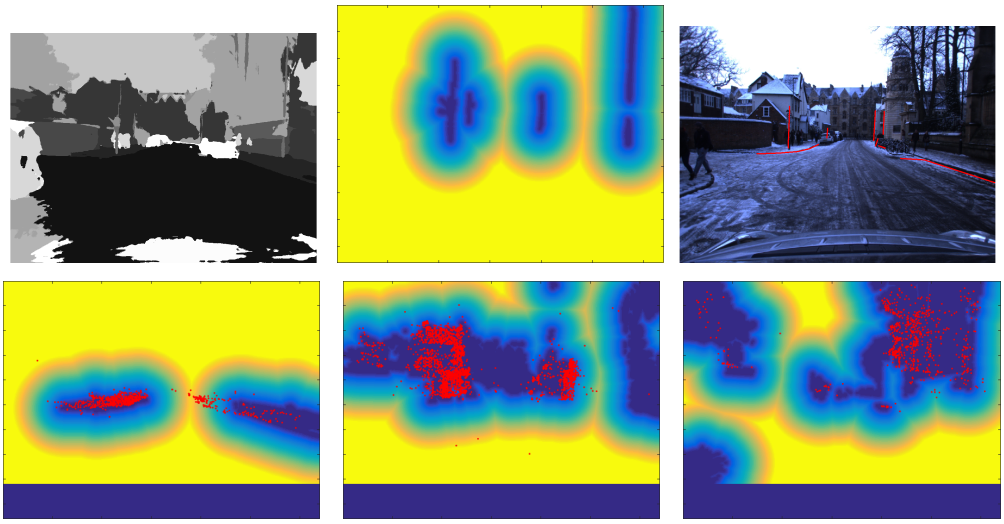


Figure 8: *Top:* An example of a successful localization from the winter sequence. The image to the top left is the semantic segmentation of the query image. The top middle image shows the error map η corresponding to the poles observed in the segmented image. The figure on the top right shows the original version of the query image before semantic segmentation, together with the 3D structure projected down into the estimated camera P found by minimizing the loss function. *Bottom:* The error maps d_{L_i} for the classes sidewalk, building and vegetation, respectively, together with the corresponding 3D points X_i projected down onto the estimated camera P.

We believe that these results are very encouraging, since it implies that the steady progress of pixelwise semantic labelling can naturally be leveraged to improve the robustness of localization algorithms that otherwise have trouble when mapping and localization occur far apart in time.

This work has been funded by the Swedish Research Council (grant no. 2016-04445), the Swedish Foundation for Strategic Research (Semantic Mapping and Visual Navigation for Smart Robots) and Vinnova / FFI (Perceptron, grant no. 2017-01942).

Bibliography

[1] D. Marr, *Vision*. W. H. Freeman and Company, 1982.

[2] F. Kahl and R. Hartley, "Multiple view geometry under the L_∞-norm", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 9, pp. 1603–1617, 2008.

[3] S. Agarwal *et al.*, "Building rome in a day", *Commun. ACM*, vol. 54, no. 10, pp. 105–112, 2011.

[4] P. Moulon, P. Monasse, and R. Marlet, "Global fusion of relative motions for robust, accurate and scalable structure from motion", in *International Conference on Computer Vision*, 2013.

[5] L. Svärm *et al.*, "City-scale localization for cameras with known vertical direction", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1455–1461, 2017.

[6] 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, no. 9, pp. 1744–1756, 2017.

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

[8] S. Lowry *et al.*, "Visual place recognition: A survey", *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.

[9] D. Lowe, "Distinctive image features from scale-invariant keypoints", *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[10] E. Rublee *et al.*, "Orb: An efficient alternative to sift or surf', in *2011 International Conference on Computer Vision*, Nov. 2011, pp. 2564–2571.

[11] H. Bay *et al.*, "SURF: Speeded up robust features", *European Conference on Computer Vision*, vol. 110, no. 3, pp. 346–359, 2008.

[12] C. Valgren and A. J. Lilienthal, "SIFT, SURF & seasons: Appearance-based long-term localization in outdoor environments", *Robotics and Autonomous Systems*, vol. 58, no. 2, pp. 149–156, 2010.

[13] K. M. Yi *et al.*, "LIFT: Learned invariant feature transform", in *European Conference on Computer Vision*, 2016.

[14] H. Badino, D. Huber, and T. Kanade, "Visual topometric localization", in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, IEEE, 2011, pp. 794–799.

[15] H. Badino *et al.*, "Real-Time Topometric Localization", in *ICRA*, 2012.

[16] M. Milford and G. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights", in *International Conference on Robotics and Automation*, 2012.

[17] M. Brubaker, A. Geiger, and R. Urtasun, "Map-based probabilistic visual self-localization", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 4, pp. 652–665, 2016.

[18] Y. Li, N. Snavely, and D. P. Huttenlocher, "Location recognition using prioritized feature matching", in *European Conference on Computer Vision*, 2010.

[19] D. M. Chen *et al.*, "City-scale landmark identification on mobile devices", in *Conference Computer Vision and Pattern Recognition*, 2011.

[20] Y. Li *et al.*, "Worldwide pose estimation using 3D point clouds", in *European Conference on Computer Vision*, 2012.

[21] S. Choudhary and P. Narayanan, "Visibility probability structure from sfm datasets and applications", in *European Conference on Computer Vision*, 2012.

[22] L. Svärm *et al.*, "Accurate localization and pose estimation for large 3D models", in *Conference Computer Vision and Pattern Recognition*, 2014.

[23] T. Sattler *et al.*, "Hyperpoints and fine vocabularies for large-scale location recognition", in *International Conference on Computer Vision*, 2015.

[24] B. Zeisl, T. Sattler, and M. Pollefeys, "Camera pose voting for large-scale image-based localization", in *International Conference on Computer Vision*, 2015.

[25] V. Larsson *et al.*, "Outlier rejection for absolute pose estimation with known orientation", in *British Machine Vision Conference*, 2016.

[26] C. Linegar, W. Churchill, and P. Newman, "Made to measure: Bespoke landmarks for 24-hour, all-weather localisation with a camera", in *International Conference on Robotics and Automation*, 2016.

[27] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization", in *International Conference on Computer Vision*, 2015.

[28] Z. Chen *et al.*, "Deep learning features at scale for visual place recognition", *CoRR*, vol. abs/1701.05105, 2017.

[29] G. Pascoe, W. Maddern, and P. Newman, "Robust direct visual localisation using normalised information distance", in *British Machine Vision Conference*, 2015.

[30] N. Atanasov *et al.*, "Semantic localization via the matrix permanent", in *Robotics: Science and Systems*, 2014.

[31] C. Häne *et al.*, "Dense semantic 3D reconstruction", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017, Accepted for publication.

[32] W. Maddern *et al.*, "1 Year, 1000km: The Oxford RobotCar Dataset", *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017. eprint: `http://ijr.sagepub.com/content/early/2016/11/28/0278364916679498.full.pdf+html`.

[33] G. Ghiasi and C. C. Fowlkes, "Laplacian reconstruction and refinement for semantic segmentation", in *European Conference on Computer Vision*, 2016.

[34] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding", in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[35] O. Enqvist, C. Olsson, and F. Kahl, "Non-sequential structure from motion", in *Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras (OMNIVIS)*, 2011.

[36] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Communications of the ACM*, 1981.