POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Visual localization in presence of match scarcity

Tesi di Laurea Magistrale in
Mathematical Engineering - Ingegneria Matematica

Author: **Valentina Sgarbossa**

Student ID: 965218
Advisor: Prof. Luca Magri
Co-advisors: Prof. Giacomo Boracchi, Dr. Antonino Maria Rizzo
Academic Year: 2021-22

Abstract

Visual localization is the problem of estimating the pose of a camera from a query image, comparing it to a 3D reconstruction of the scene. When the scene was obtained in very different visual conditions from the query, e.g. seasonal changes, we talk about *long-term* localization. Applications of this problem are already part of our lives, from autonomous driving to augmented reality.

Visual localization usually involves matching points of the query to those in the 3D structure, and using these matches to infer a pose through geometric reasoning. In the long-term setting, matching is challenged by the mutated appearance of keypoints, causing *match scarcity*. The subsequent pose estimation produces lower quality, or even completely wrong, pose estimates.

In this thesis, we propose solutions to the problem of localizing with match scarcity, through fine-grained segmentations robust to long-term visual variations. Our contribution is two-fold: (*i*) we devise a novel matching strategy – *Semantic Matching* – which combines local appearance information to global semantic cues to provide higher quality matches, (*ii*) we modify the random sampling and consensus pose estimation algorithm to account for models with few inliers with high semantic consistency, as it happens with match scarcity. The resulting method, *Biased Consensus*, selects the sampled model with largest overall semantic consistency.

We experimentally verify that Semantic Matching improves the quality of matches and increases their quantity. Moreover, it shows excellent performances in the task of pose estimation, outperforming state-of-the-art methods on sequences with match scarcity on all accuracy levels. The combination of Semantic Matching and Biased Consensus exhibits more than doubled correct localizations on the best performing method from literature.

Keywords: visual localization, camera pose estimation, semantic consistency, robust fitting

Abstract in lingua italiana

La localizzazione visiva consiste nello stimare la posa di una fotocamera da un'immagine query, comparandola a una ricostruzione 3D della scena. Se la scena osservata nella ricostruzione è radicalmente diversa dalla query, ad esempio in presenza di variazioni stagionali, si parla di localizzazione a lungo termine. Applicazioni di questo problema sono già parte della quotidianità, dalla guida autonoma alla realtà aumentata.
La localizzazione visiva solitamente segue due fasi. In primo luogo, punti della query vengono accoppiati a punti della ricostruzione 3D. Quindi le corrispondenze vengono usate per ricavare una posa tramite ragionamenti geometrici. Nello scenario di lungo termine, la creazione di corrispondenze è messa alla prova dai cambiamenti di aspetto dei punti descritti, e la conseguente scarsità di corrispondenze.
In questa tesi, proponiamo una soluzione al problema di localizzazione con scarsità di corrispondenze attraverso segmentazioni fini delle immagini. Il nostro contributo si articola in due strumenti: (*i*) una nuova strategia per la creazione di corrispondenze – *Semantic Matching* – che combini informazione locale sull'aspetto di porzioni di immagine con segnali semantici globali, per fornire corrispondenze di alta qualità, (*ii*) una nuova versione dell'algoritmo di stima robusta della posa così da tenere conto di modelli con pochi inliers, ma complessivamente fortemente consistenti dal punto di vista semantico. Il metodo risultante, *Biased Consensus*, seleziona la posa con consistenza semantica maggiore.
Verifichiamo sperimentalmente che Semantic Matching migliora sia la qualità, sia la quantità di corrispondenze trovate. Inoltre, i risultati della stima della posa sono eccellenti, e in contesti con scarsità di match superano per ogni soglia di accuratezza algoritmi allo stato dell'arte. Infine, la combinazione di Semantic Matching e Biased Consensus permette di raggiungere oltre il doppio di immagini correttamente localizzate rispetto al miglior metodo dello stato dell'arte.

Parole chiave: localizzazione visiva, stima della posa, consistenza semantica, fitting robusto

Contents

1 | Introduction

1.1. The problem of visual localization

Visual localization is the problem of inferring the pose of a camera from a picture, the *query*. This tool has several applications in contemporary technologies, from autonomous driving to augmented reality, which require stable and reliable information about the location of cars, or devices, in space. Doing so with a standard camera rather than with the help of ad-hoc sensing tools, such as LIDAR, is a significant step towards lightweight and cost-effective localization.
Differently from visual place recognition, whose focus is on recognizing the broader location of a picture, visual localization entails accurate identification of the position and orientation of the camera. To obtain such precise information, the query is compared to a 3D reconstruction of the scene, previously obtained from a database of images. Correspondences are found between 2D interest points in the image and 3D points. Once several of these correspondences have been found, a pose can be estimated via geometric reasoning. To contrast the negative effect on the pose of incorrect matches, a robust estimation framework, like the Random Sample Consensus (RANSAC) algorithm [13] is employed. In Figure 1.1 we represent visual localization.

Although some outliers can be tolerated with robust model estimation, the higher the percentage of these, the harder, and more computationally intensive, the estimate. The problem is even worsened by scenes evolving in time, where corresponding points disappear or move in relation to one another, and the appearance of objects varies. Think, for example, of a tree observed across seasons. The foliage, as well as the colour and size of the tree are subject to variations. It is highly desirable to minimize the impact of these variations on the localization performances.
Because traditional, appearance-only based localization methods [34, 42] have difficulties reaching high precision estimates in long-term localization settings [37], literature has focused on techniques to complement appearance information, so to achieve robustness even when scenes are visually different. Semantics are well suited to this goal, because they describe the nature of objects and not their appearance. For example, one can verify

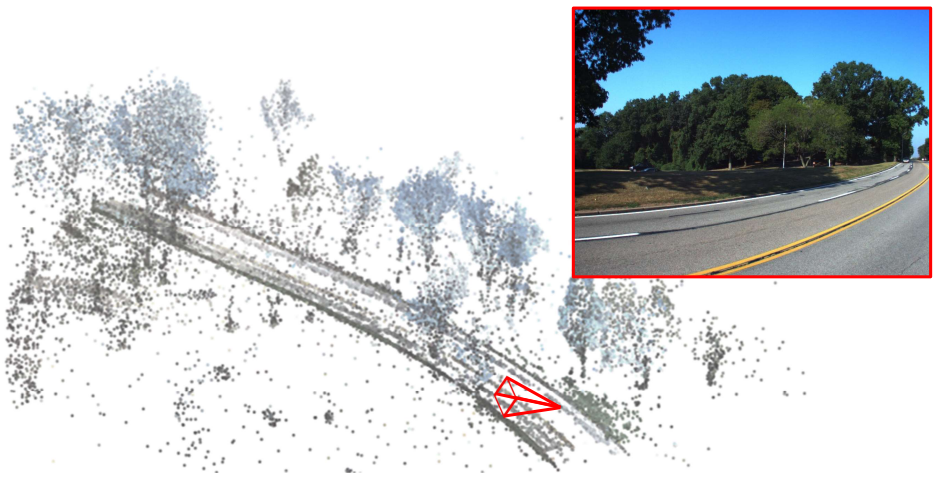


Figure 1.1: Visual localization: a query is registered to a point cloud of the scene.

if matching 2D and 3D points are also representing *consistent* semantic classes, hence discarding accidental matches arising from similarly visual patches of completely different objects.

One successful approach employs semantic consistency of a match to the scene as a soft filter, and promotes the sampling of the most promising matches during pose estimation [44]. Furthermore, a finer and more robust segmentation can be obtained through self-supervision [22], so that the score becomes even more discriminative. However, whereas these methods increase the chance to pick the correct model during pose estimation, they still inherently depend on the quality of matches that are presented to the filter. Since this is only assessed through appearance, long-term variations may disrupt a significant portion of matches before the semantic filter comes in, leaving few valid matches overall (*match scarcity*). Hence, some effort should be made to improve the quality of matches while these are created. In the context of urban localization, some authors [35, 49] proposed to tackle the ambiguity of repetitive structures across the scene by exploring multiple match options for all query keypoints.

The first finding of this work is that the same strategy can be successfully applied not only to man-made repeated structures, but also to a broader range of scenarios with long-term variations. Indeed, as appearance changes there is higher probability to find ambiguity of descriptors, and thus to find the correct descriptor within the first few neighbors, although

not necessarily the first.
We furthermore observe that localizing with match scarcity entails worse pose estimates. Indeed, the underpinning assumption of robust pose estimation is that the correct pose will have a larger *consensus* than other poses obtained with incorrect match samples, that is the number of matches supporting the correct pose will always exceed that of other randomly sampled poses. However, in the presence of match scarcity the overall amount of correct matches could lower significantly, becoming comparable to the achieved consensus of random poses. Thus, we verify the need to re-design the evaluation of poses during robust fitting.

1.2. Our contribution

In the present work we make the following contributions:

(i) we demonstrate the benefits of shifting the focus of semantic consistency from a post-matching filtering stage to the matching procedure itself, when facing situations with ambiguous matching and match scarcity;

(ii) we devise a framework to perform *Semantic Matching*, revisiting the Geometric-Semantic Match Consistency algorithm of [44];

(iii) we propose an ad-hoc pose estimation variation of the RANSAC [13] algorithm, adding *Biased Consensus*, which allows correct pose estimation with severe match scarcity.

We perform experiments on the Extended CMU Seasons dataset [4, 37] to analyse in depth our method. Finally, we compare the proposed versions to relevant localization pipelines directly on the task of pose estimation.

1.3. Outline

The rest of this work is organized as follows. In chapter 2 we report theoretical notions from computer vision and projective geometry as background to all our work. An expert reader may skip this part.
In chapter 3 we present the problem formulation, and discuss relevant work in chapter 4. We propose our method in chapter 5, whose validity is extensively verified in chapter 6. Finally, in chapter 7 we summarize findings and discuss interesting future research lines.

2 | Domain background

As introduced in the previous chapter 1, being able to estimate the location from which an image was captured is of paramount importance for a variety of applications. Although assumptions and problem boundaries vary with the application, at the heart of the problem is a geometric modeling of the camera. In this chapter we present the mathematical foundations of this theory, giving the necessary background for the following work.

2.1. Camera geometry

When capturing a picture, both analog and digital photography start with transferring the appearance (in the form of light) of a 3D scene to a bi-dimensional screen in a process called image formation. In reality, light is captured by a system of lenses, which aim to maximize the amount of light that reaches film or CCD (Charge Coupled Device, the device that transforms light stimuli in electrical signals for digital cameras). For our purposes, a simplified model of the camera will however suffice. In this simpler representation, rays of light go through a small hole surrounded by an opaque screen for the image to be correctly rendered (inverted) on screen. Such device is called pinhole. Figure 2.1 illustrates this idea.

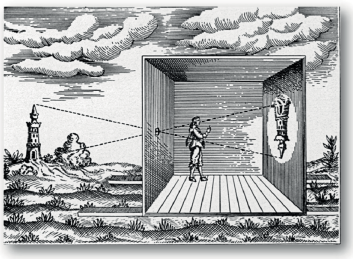


Figure 2.1: The process of image formation for a pinhole camera. Figure from [16].

The geometric transformation from 3D to 2D points is also named *perspective projection*. It consists in projecting points $\mathbf{X} \in \mathbb{R}^3$ to $\mathbf{x} \in \mathbb{R}^2$, as outlined in Figure 2.2. A point C, representing the *camera centre*, is the origin of a 3D reference system for points in the

world. The centre is the ideal location of the pinhole, while the image will be projected onto the *image plane* (physically, the film or CCD). The *principal axis* is the orthogonal line to the image plane through the camera centre. Note the z axis of the 3D reference system is set parallel to this line. The intersection of the principal axis and image plane is the *principal point*, on which the origin for a 2D image reference system is commonly set. Finally, the distance between camera centre and image plane is the *focal length* f.

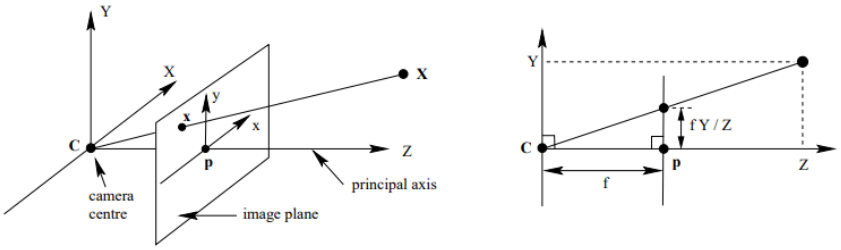


Figure 2.2: Pinhole geometry outline. Figure from [15].

Thanks to similarities among triangles, the projection of $\mathbf{X} = (x, y, z)^T$ onto the image plane has well-defined coordinates $\mathbf{x} = (fx/z, fy/z)^T$.
The final image representation requires a further change of coordinates, from physical CCD elements (*pixels*) to image coordinates. In particular, the conversion possibly entails a shift of the origin, so the principal point is mapped to (p_x, p_y). Moreover, as pixels might not be square, a relative rescaling of the abscissa and ordinate in the image reference system could be needed. The factor to correct for this is called *aspect ratio*, and is usually defined as $r = s_x/s_y$. After applying this rescaling, the principal point is mapped to (u_0, v_0). It could also happen that the CCD elements are skewed from rectangular form to parallelogram form. Such a deformation is then expressed by the skew γ.

All together, these allow a compact representation of the perspective projection in matrix form, provided homogeneous coordinates are introduced.

2.2. Projective geometry

We start by observing that a projective transformation maps 3D lines to 2D lines, but does not necessarily preserve parallelism of two 3D lines. Indeed, we can think of a picture of straight railroad. Only if the railroad runs parallel to the horizon line, we will correctly project the binaries as parallel. In all other cases, binaries will appear to converge to a unique point at infinite distance. Such point has well defined location in the image, and yet has no 3D correspondent.
Motivated by this example, we introduce a larger vector space to hold both points at

finite and at infinity. A projective space $\mathbb{P}^3$ is obtained by extending $\mathbb{E}^3$, the Euclidean 3D space, with a plane at infinity, i.e. the locus of improper points. An improper point is the ideal intersection of parallel lines with fixed direction.
Points in a 3D projective space are conveniently represented with 4 homogeneous coordinates, namely they are 4-tuples defined except for a multiplicative factor. Let, for example, $\boldsymbol{X} = (x,\ y,\ z)^T \in \mathbb{E}^3$ represent a point in space.
The corresponding homogeneous point is $\tilde{\mathbf{X}} = \begin{pmatrix} s\mathbf{X} \\ s \end{pmatrix} \in \mathbb{P}^3,\ s \neq 0$, which can then be normalized $\tilde{\mathbf{X}} = \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$.
The original point in $\mathbb{E}^3$ can be retrieved by taking the first three coordinates and dividing them by the last one.
Improper points have instead representation $\tilde{\mathbf{X}} = (x,\ y,\ z,\ 0)^T$, which does not have a corresponding point in $\mathbb{E}^3$. Indeed, intuitively, if we were to divide the first three coordinates by the last one, the resulting point would degenerate to infinity in the direction of $\mathbf{X} = (x,\ y,\ z)^T$.

One key advantage of using homogeneous coordinates is that they unify the representation of transformations such as rotations, translations and similarities in the concept of projective map. General projective transformations are named *collineations* or *homographies*, and are represented by a 4 x 4 invertible matrix. Among these we will be especially concerned with affine rigid transformations with proper rotation (thus excluding reflections).

Definition 1. The special Euclidean group $SE(n)$ is the set

$$SE(n) = \{A \mid A = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times n} & 1 \end{bmatrix},\ \mathbf{R} \in \mathbb{R}^{n\times n},\ \mathbf{t} \in \mathbb{R}^n,\ \mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = 1\}. \tag{2.1}$$

Transformations in $SE(3)$ are known as *6 DoF poses*, because they have 6 free parameters (or *Degrees of Freedom*).

Finally, to project , the 3D world to a 2D image representation *central projection* will be used.

2.3. Perspective projections

We now identify the representation of a projective map from a point $\mathbf{X}_W \in \mathbb{P}^3$ to a point $\mathbf{x}_I \in \mathbb{P}^2$, where W indicates the reference system of the world, and I the pixel coordinates

reference system.

A generic transformation consists in the following steps:

1. From a 3D point in the world reference system, map the point to its coordinates in the camera reference system. This is represented by a matrix $\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1xn} & 1 \end{bmatrix} \in SE(3)$.
2. Project the point from 3D to 2D on the image plane. This is represented by the matrix $M = diag(f, f, 1)[I|\mathbf{0}] = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$.
3. map physical coordinates to pixel coordinates in 2D. As described in Section 2.1, this entails skewing, rescaling and translating the image coordinate system. A matrix representation for this is $\tilde{K} = \begin{bmatrix} 1 & \gamma & u_0 \\ 0 & r & v_0 \\ 0 & 0 & 1 \end{bmatrix}$ where $\gamma \approx 0$ in most applications.

The first step is concerned with the position of the camera in the world. These are defined *extrinsic parameters* of the camera. Conversely, the second and third step are influenced by the internal design of the camera, the *intrinsic parameters*.
It is often convenient to rearrange the representation for the whole transformation in the composition of two matrices, one for the intrinsics and one for the extrinsics. We thus obtain the 3 x 4 *camera projection matrix* P:

$$P = K[R|\mathbf{t}] = \begin{bmatrix} f & \gamma f & u_0 \\ 0 & rf & v_0 \\ 0 & 0 & 1 \end{bmatrix} [R|\mathbf{t}]. \tag{2.2}$$

The matrix K is called *camera calibration matrix*. It is often the case that this matrix is known, and in such circumstances the camera is said to be *calibrated*. We will later see how calibration plays an important role in removing projective ambiguities from the reconstruction, and therefore it is very important to have access at least to partial calibration information. Fortunately, most images nowadays contain metadata regarding camera calibration.

Finally, the transformation is summarized as

$$z\tilde{\mathbf{x}}_I = P\tilde{\mathbf{X}}_W \tag{2.3}$$

with $\tilde{\mathbf{x}}_I = (x_I,\ y_I,\ 1)^T$ and $\tilde{\mathbf{X}}_W = (x_W,\ y_W,\ z_W,\ 1)^T$.

We will from now on omit the subscripts I and W, yet still refer to mappings from a unified world reference system to individual camera coordinates.

2.4. Camera anatomy

We now analyze a few concepts of camera geometry in light of the projective geometry framework we have just introduced.

Ray of a point. The ray of a 3D point is the line through the point itself and the camera centre. It is mapped on a unique point in the image, and we can therefore identify the central projection as a mapping of rays (lines) of $\mathbb{P}^3$ to points of $\mathbb{P}^2$. Thanks to the introduced generalization, points at infinity are treated just as any other point in the ray. In particular, the horizon line is visible in the image through this mechanism.

The generic equation for a ray through the point $\tilde{\mathbf{A}} \in \mathbb{P}^3$ is

$$\tilde{\mathbf{X}}(\lambda) = \lambda\tilde{\mathbf{A}} + (1-\lambda)\tilde{\mathbf{C}} \tag{2.4}$$

and its mapping to $\mathbb{P}^2$ is then

$$\tilde{\mathbf{x}} = \lambda P\tilde{\mathbf{A}} + (1-\lambda)P\tilde{\mathbf{C}}. \tag{2.5}$$

Camera centre. The only 3D point whose mapping is undefined in the image is the camera centre. Indeed, by definition the centre of projection is the point that belongs to every ray. Intuitively, such point has no defined direction.
More formally, we select the point $\tilde{\mathbf{C}}$ such that $P\tilde{\mathbf{C}} = 0$ and show it coincides with the camera centre, as every ray passes through this point.
Consider a generic $\tilde{\mathbf{A}} \in \mathbb{P}^3$. We specialize equation 2.3 to $\tilde{\mathbf{x}} = \lambda P\tilde{\mathbf{A}}$ (where the multiplicative factor has been moved to the right side for convenience) and add the null term $(1-\lambda)P\tilde{\mathbf{C}}$. We finally obtain an equation in the form of 2.5, meaning the point $\tilde{\mathbf{C}}$ belongs to the ray of $\tilde{\mathbf{A}}$. For arbitrariness of $\tilde{\mathbf{A}}$ we conclude $\tilde{\mathbf{C}}$ is the centre of projection.

Assuming P is full rank, we may recover the Euclidean coordinates of the centre of projection $\mathbf{C}$ from knowledge of P. Indeed,

$$0 = P\tilde{\mathbf{C}} = P_{,1:3}\mathbf{C} + P_{,4}$$

thus obtaining

$$\mathbf{C} = -P_{\cdot,1:3}P_{\cdot,4}$$

where we denoted with $P_{\cdot,1:3}$ the first three columns of P, and $P_{\cdot,4}$ is the last column of P. Since the matrix K of camera internals is upper triangular, thus invertible, we may also rewrite the relation between camera centre and translation vector as

$$\mathbf{C} = -R^T\mathbf{t}. \tag{2.6}$$

Depth of a point. The multiplicative factor z appearing in equation 2.3 has an interesting interpretation. This factor depends on the representation that is chosen for P. Indeed, since 2.3 is a homogeneous equation, any P defined up to a multiplicative factor maps $\tilde{\mathbf{X}}$ to the same point in $\mathbb{E}^2$.
Considering homogeneous representations of $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{C}}$, we may assume both have last coordinate 1.

Then

$$z\tilde{\mathbf{x}} = P\tilde{\mathbf{X}} = P(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}) = P_{\cdot,1:3}(\mathbf{X} - \mathbf{C})$$

where we have denoted with $P_{\cdot,1:3}$ the first three columns of P. According to the decomposition of P introduced in 2.2, the third row of $P_{\cdot,1:3}$ coincides with the third row of the rotation matrix R. Therefore, such vector represents the direction of the principal axis. By taking the third equation, we find

$$z = P_{3,1:3}(\mathbf{X} - \mathbf{C}).$$

Thus, if the matrix P is scaled such that $K_{3,3} = 1$, z represents the projection of the vector $\mathbf{X} - \mathbf{C}$ onto the principal axis, thus the *depth* of $\mathbf{X}$.

2.5. Absolute pose solving

The final objective of our efforts will be to register effectively a query image to the world coordinates. We now assume sufficient correspondences $\{(\mathbf{X}_i, \mathbf{x}_i)\}$, $\mathbf{X}_i \in \mathbb{R}^3$, $\mathbf{x}_i \in \mathbb{R}^2$, $i = 1, ..., N_c$ are given, and see how P can be estimated. From time to time we will refer to this problem as *absolute pose estimation, image registration* or simply *pose estimation.*
Let us start from the *uncalibrated* case, where we have complete uncertainty about matrices K, R and $\mathbf{t}$.
In this general case, assuming the matrix P is full rank, the solution is found with the

Direct Linear Transform (DLT) algorithm.

2.5.1. DLT equations

For every match $(\mathbf{X}_i, \mathbf{x}_i)$, it must hold equation 2.3. The equation entails vectors $\mathbf{x}_i$ and $P\mathbf{X}_i$ are parallel, or their vector product is null. We may rewrite it conveniently by adopting the skew-symmetric representation of $\mathbf{x}_i$:

$$[\mathbf{x}_i]_x P\mathbf{X}_i = 0 \iff vec([\mathbf{x}_i]_x P\mathbf{X}_i) = 0 \iff (\mathbf{X}_i^T \otimes [\mathbf{x}_i]_x) vec(P) = 0$$

and finally obtain a system of 3 equations in 12 unknowns, with 11 DoF, since P is unique up to a scale factor:

$$\begin{bmatrix} \mathbf{0}^T & -x_{i,3}\mathbf{X}_i^T & x_{i,2}\mathbf{X}_i^T \\ x_{i,3}\mathbf{X}_i^T & \mathbf{0}^T & -x_{i,1}\mathbf{X}_i^T \\ -x_{i,2}\mathbf{X}_i^T & x_{i,1}\mathbf{X}_i^T & \mathbf{0}^T \end{bmatrix} \begin{pmatrix} P_{1,\cdot}^T \\ P_{2,\cdot}^T \\ P_{3,\cdot}^T \end{pmatrix} = 0.$$

These equations are linearly dependent. In fact, $[\mathbf{x}_i]_x$ has rank 2, and the Kronecker product of a rank 1 matrix by a rank 2 matrix yields a rank 2 matrix. By convention, the third equation is dropped, yielding the final system:

$$\begin{bmatrix} \mathbf{0}^T & -x_{i,3}\mathbf{X}_i^T & x_{i,2}\mathbf{X}_i^T \\ x_{i,3}\mathbf{X}_i^T & \mathbf{0}^T & -x_{i,1}\mathbf{X}_i^T \end{bmatrix} \begin{pmatrix} P_{1,\cdot}^T \\ P_{2,\cdot}^T \\ P_{3,\cdot}^T \end{pmatrix} = 0. \tag{2.7}$$

Therefore, each point correspondence adds two equations, and a total of 5 $\frac{1}{2}$ correspondences suffice to solve for P.
Overall, the system may be written as $A\mathbf{p} = 0$, where A is the matrix obtained by stacking the equations for 5 $\frac{1}{2}$ correspondences, and $\mathbf{p} = vec(P)$.
In most cases, the correspondences are not exact but noisy. Thus, it is desirable to use a least-square generalization of the system. One way to proceed is to minimize $\|A\mathbf{p}\|$ subject to $\|\mathbf{p}\| = 1$.
Finally, the DLT does not generally yield a P that is consistent with any assumptions on the camera internals. To add any domain information, different procedures are needed.

2.5.2. The perspective-n-point problem

The *perspective-n-point* problem (PnP) is an alternative formulation of the image registration problem in the assumption of calibrated camera.

Given a set of n points in space and their location on an image, the problem is to find the length of the segments from each 3D point to the centre of perspective. With a sufficient number of these points, the centre and orientation of the camera are determined without ambiguity.

We now outline the solution to the P3P problem. We start by considering three control points $A, B, C \in \mathbb{R}^3$. The centre of projection C_P forms with these points a tetrahedron, whose vertex we take as the C_P itself. The base of the tetrahedron is fully known – angles and sides, which we will name R_{ab}, R_{bc}, R_{ac} –, as well as the angles $\theta_{ab} = \widehat{AC_PB}$, $\theta_{ac} = \widehat{AC_PC}$, $\theta_{bc} = \widehat{BC_PC}$ (cf. Figure 2.3). Note that the image projections of the control points $\mathbf{x}_A$, $\mathbf{x}_B$, $\mathbf{x}_C$, are not considered in this problem, but they are important to determine the face angles θ_{ab}, θ_{bc}, θ_{ac}. Indeed, the same angles are formed in the faces of the tetrahedron with base the triangle of $\mathbf{x}_A$, $\mathbf{x}_B$, $\mathbf{x}_C$ on the image plane and with vertex C_P. This tetrahedron is fully determined because the camera calibration is known.

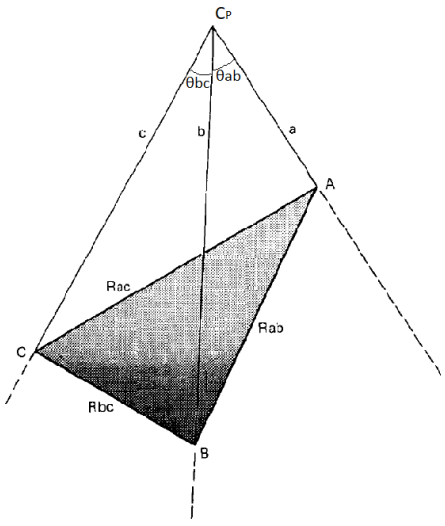


Figure 2.3: Geometry of the P3P problem. Figure adapted from [13]

The objective is to determine the lengths of the sides a, b, c of the tetrahedron ("legs").

One can write three equations by using the law of cosines:

$$R_{ab}^2 = a^2 + b^2 - 2ab\ cos\ \theta_{ab}$$
$$R_{bc}^2 = b^2 + c^2 - 2bc\ cos\ \theta_{bc}$$
$$R_{ac}^2 = a^2 + c^2 - 2ac\ cos\ \theta_{ac}.$$

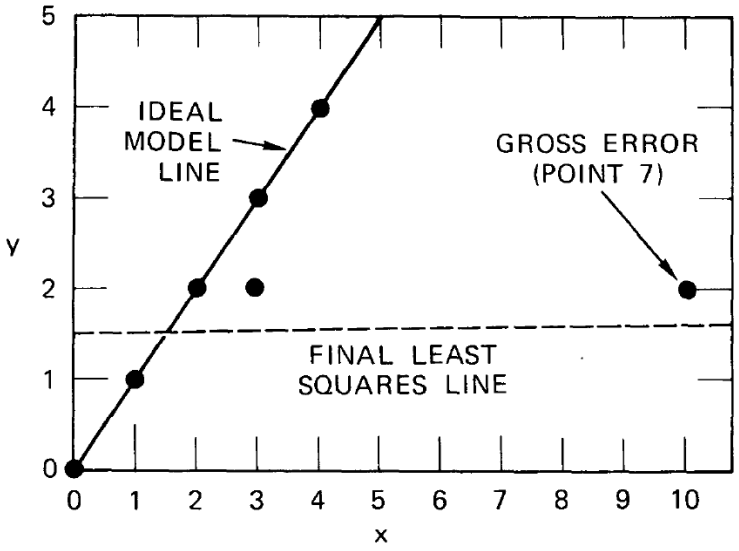
[13] show that this system of equations has no more than 4 admissible solutions, depending on the configuration of the problem. Interestingly, when extending to more than 3 control points, for 4 and 5 correspondences there is still no guarantee of a unique solution, unless the points are coplanar. When providing 6 or more point correspondences, instead, a unique solution can always be found.

In practice, since a robust estimation scheme has to be used and different models have to be taken in consideration, the preferred approach is to proceed with P3P and disambiguate among the 4 solutions through consensus.

The above procedure, however, considers an exact geometry, and no errors are accounted for in the matches. In reality, errors are frequent and both imprecise measurement and false matches can hinder the estimate. In light of this, robust estimation is crucial for success.

2.6. RANSAC

We present here the general framework of the RANdom SAmple Consensus (RANSAC) algorithm, first introduced by [13] in the domain of pose estimation, but that can be used in greater generality for robust fitting in a range of problems where gross errors may occur. In visual localization the scheme is used, for instance, for absolute and relative pose estimation and pose verification.



POINT	x	y
1	0	0
2	1	1
3	2	2
4	3	2
5	3	3
6	4	4
7	10	2

Figure 2.4: An example of failure of a heuristic outlier detector. The procedure consists in fitting a model through least squares, and later removing any points that deviate excessively from the model. In this configuration, the model is very close to point 7, which is in reality an outlier. Point 6 is instead labelled as outlier, despite being consistent with the "real" model. Figure from [13].

Many model fitting procedures are based on sample mean as an estimator. It is well known that this estimator is not robust, since very large extreme sample values can impact severely on the resulting mean. Let us consider, as an example, fitting a line to some data with ordinary least squares (cf. Figure 2.4).
The idea behind RANSAC is quite straightforward: rather than attempting to fit a model to all the available data and removing outliers only after the fitting – when the model might be already compromised by outliers – several models are estimated with few points and the one with the largest *consensus* within data is eventually chosen. In other words, RANSAC is a framework for joint model fitting and selection, outputting both a model and a subset of data that supports that model.

Algorithm 2.1 RANSAC

Inputs: candidate points set S, maximum number of iterations N_{iter}, inlier threshold ϵ

Output: model θ^*, inliers S^*

best model $\theta^* = N.A.$, optimal consensus set $CS(\theta^*, \epsilon) = \emptyset$
for all $k = 1, ..., N_{iter}$ **do**
  Sample a random minimal set S_k from S
  Fit model θ_k to S_k
  if $|CS(\theta_k, \epsilon)| > |CS(\theta^*, \epsilon)|$ **then**
    $\theta^* = \theta_k$
  end if
end for
Fit model to $CS(\theta^*, \epsilon)$
=0

Let us describe in more detail RANSAC's algorithm, as outlined in Algorithm 2.6.
Let S be the dataset. The fitting process starts by randomly sampling a minimal set of data points $S_k \subset S$ to fit a model θ_k. In our example of linear regression, S_k would be two points, and θ_k the slope and intercept of the line. Such model is then evaluated based on its *consensus set* $CS(\theta, \epsilon)$. Usually, we count how many points are within some predefined error ϵ, the *inlier threshold*, from the estimated model, so in our example we would draw a band around the fitted line and count how many data points are included within that region. Sampling and model fitting are repeated for a fixed number of iterations N_{iter}, and eventually the model θ^* with largest support is returned, along with its consensus set $CS(\theta^*, \epsilon)$. One could even specify an absolute threshold τ on the consensus set size, such that if the consensus is larger, the algorithm terminates before completing all iterations. Note that, if unlimited resources were available, one could try all possible models and return the best – RANSAC would therefore deterministically choose the model with the largest consensus. However, since the number of possible models is way too large to be searched exhaustively, a probabilistical solution is adopted. Thus, the maximum number of iterations of RANSAC is usually decided by fixing the probability of not finding the optimal solutions by that number of attempts. This can be easily done if an estimate of the probability p for a data point to be an outlier is available, and under the assumption that the "inlier-ness" of point is independent on that of other points, which might not always hold. In this framework, the probability to sample at least one model with inliers only by the k-th iteration is $1 - (1 - (1 - p)^m)^k$, hence to have such probability larger

than a fixed probability α, we have

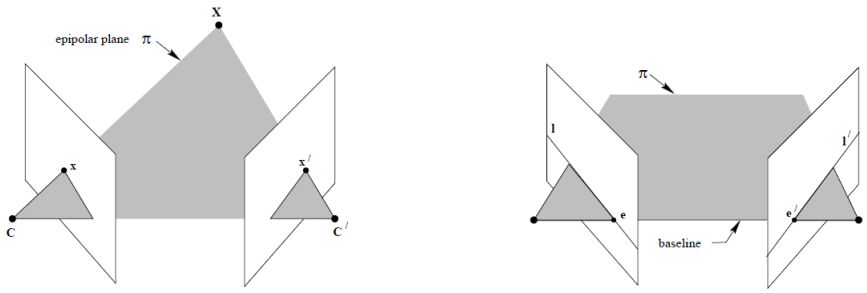
$$N_{iter} \approx \frac{log(1-\alpha)}{log(1-(1-p)^m)}. \tag{2.8}$$

Naturally, values of α close to 1 will demand a high confidence on the chosen model, thus leading to larger iteration numbers, while values close to 0 will accept nearly any model.

Experimentally, the results of RANSAC are outstanding, as it can handle accurately datasets with a majority of outliers.

2.7. Epipolar geometry

In the following subsection we explore the main results in the two-view setting, where two images observing the same scene are given, but the 3D structure is not provided. The images can be taken simultaneously from slightly different points of view, or in subsequent moments for a moving camera, provided that the scene can be approximated as unchanged.
The broader goal of two-views geometry is to use 2D-2D correspondences to jointly perform inference on camera projection matrices of both images and the location of 3D points of the scene, and is useful in visual localization to obtain a 3D reconstruction of the scene (cf. Section 4.3.1).
Our formulation, assumes that a number of 2D-2D correspondences $\{(\mathbf{x}_i,\ \mathbf{x}'_i) \mid \mathbf{x}_i \in \mathbb{R}^2,\ \mathbf{x}'_i \in \mathbb{R}^2,\ i = 1, ..., N\}$ is given, and we are interested in the camera projection matrices P, P' and the 3D location of the points $\{\mathbf{X}_i \mid \mathbf{X}_i \in \mathbb{R}^3,\ i = 1, ..., N\}$ which generated both sets of projections.
We start by analyzing epipolar geometry, the geometrical relationship between two distinct cameras that observe a common scene. Interestingly, the epipolar geometry is independent of the scene structure, and is rather a function of the cameras internals and relative position. The mapping is linear: a 3×3 matrix F (called the *fundamental matrix*) is used to describe the relationship between any two correspondences $(\mathbf{x}, \mathbf{x}')$.



(a) Epipolar geometry outline (b) Formation of the epipolar line.

Figure 2.5: (a) displays the projection of a common 3D point onto the image planes for two cameras with centers $\mathbf{C}$ and $\mathbf{C}'$. (b) illustrates the constraints of the position of a point if the centers and position in the other image are known. Figure from [15].

Figure 2.5 illustrates the main features of epipolar geometry. Two centers $\mathbf{C}$, $\mathbf{C}'$ represent the two cameras. A plane $\boldsymbol{\pi}$ called *epipolar plane* is defined for every 3D point $\mathbf{X}$ observed by both images. Thus, before observing the scene, the epipolar geometry is an *epipolar pencil*, or a pencil of planes through a common line denominated *baseline*. The centers $\mathbf{C}$, $\mathbf{C}'$ define the baseline, and the intersections of such line with the image planes are the *epipoles*. Figure 2.5b shows that fixing a point in one of the two images constrains the corresponding point in the other image to lay on a line, as these points must be coplanar. The line is called *epipolar line*, and is exemplified by $\mathbf{l}$ for the point $\mathbf{x}$' of Figure 2.5a, and by $\mathbf{l}$' for the point $\mathbf{x}$.
The epipolar geometry therefore defines a function $\mathcal{T} : \mathbf{x} \mapsto \mathbf{l}$', which depends on the configuration of the cameras and their intrinsic parameters. It is a projective transformation from points of $\mathbb{P}^2$ to lines in $\mathbb{P}^2$. The transformation can be conveniently decomposed in two steps, the first mapping $\mathbf{x}$ to a candidate $\mathbf{x}' \in \mathbf{l}'$ and in a second stage defining $\mathbf{l}$' as the line passing through $\mathbf{x}$' and the epipole $\mathbf{e}$'. It can be argued that the global transformation is represented by a 3×3 matrix F, since it is a composition of homographies.
The first transformation is a 2D homography because it preserves lines. To see this, we can think of the two views geometry as in Figure 2.5. Let ψ be a plane not containing the camera centres (see Figure 2.6). This plane intersects all rays through a camera centre in exactly one point (thereby fixing the homogeneous scale and removing any uncertainty on the pre-image of projected points). If we imagine to intersect the plane ψ with rays of a number of points that lay on a line in the image with camera $\mathbf{C}$, we find a line, since a plane ζ contains all the rays. Therefore, a plane ζ' can be found containing the intersection of ζ and ψ and the centre $\mathbf{C}$'. Finally we intersect ζ' with the image plane relative to $\mathbf{C}$' and find a line.

Overall, this first step may be formalized as follows:

$$\mathcal{T}_1 : \mathbf{x} \mapsto \mathbf{x}' = H_\psi \mathbf{x}.$$

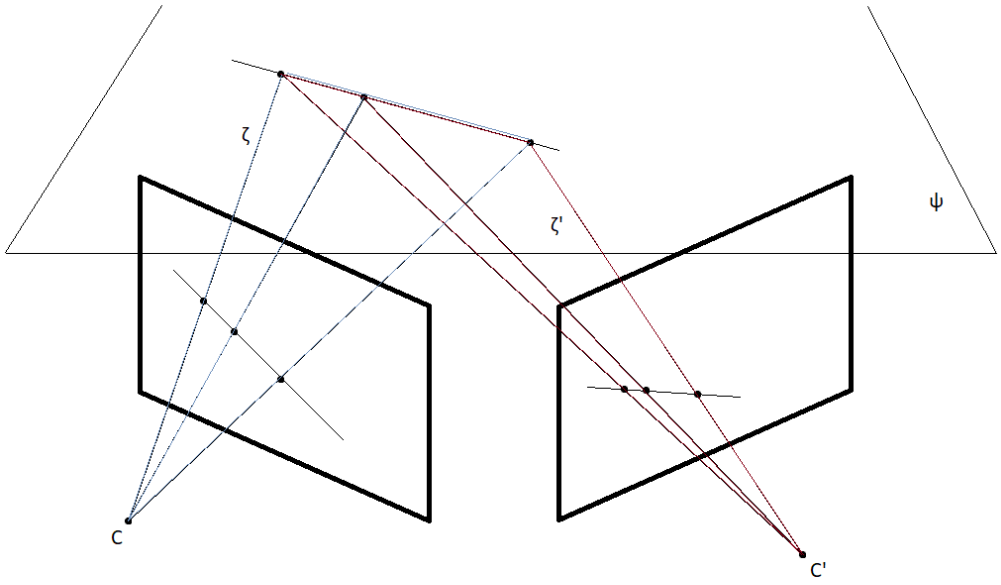


Figure 2.6: Illustration of the homography connecting points in two views.

It is also quite straightforward to map the point $\mathbf{x}$' to the line $\mathbf{l}$'. Indeed $\mathcal{T}_2 : \mathbf{x}' \mapsto \mathbf{l}' = \mathbf{e}' \times \mathbf{x}'$ since in $\mathbb{P}^2$ such relation holds in general.

By using the skew-symmetric operator, it is possible to rewrite the transformation $\mathcal{T}_2$ as a linear function of $\mathbf{x}$': $\mathbf{l}' = [\mathbf{e}']_\times \mathbf{x}'$. Finally, we obtain

$$\mathcal{T} : \mathbf{l}' = [\mathbf{e}']_\times H_\psi \mathbf{x}. \tag{2.9}$$

A few observations are now due. First, we define the fundamental matrix F as $F = [\mathbf{e}']_\times H_\psi$. H_ψ depends on the choice of ψ. Solving such ambiguity corresponds to choosing a coordinate frame in the projective space. Therefore, the matrix F is defined up to an arbitrary projective transformation.
F is also a singular matrix. Indeed, it has a zero right eigenvalue in correspondence of

the epipole **e**. To see this, it suffices to recall that every line $\mathbf{l} = F^T\mathbf{x}'$ goes through the epipole **e**. As a consequence, $\mathbf{e}^T(F^T\mathbf{x}') = (F\mathbf{e})^T\mathbf{x}') = 0$ for all **x**', which means $F\mathbf{e} = \mathbf{0}$. Therefore F has rank 2 and it is a homogeneous matrix with 7 degrees of freedom.
Finally, a more explicit expression for F is

$$F = [\mathbf{e}']_\times P'P^-, \tag{2.10}$$

where P^- is the pseudoinverse of the camera matrix P, namely $PP^- = I$.
The transformation 2.7 induces the condition

$$\mathbf{x}'^T F\mathbf{x} = 0. \tag{2.11}$$

This condition holds for every valid pair (**x**, **x**') and allows to retrieve F in case the correspondences are given.
In case the calibration is known for the cameras, a simpler expression is available, and the uncertainty on the matrix (now called *essential matrix* is reduced to a similarity (transforming rigidly and fixing a scale). It can be written as

$$E = (K')^T F K. \tag{2.12}$$

Estimating F can be done similarly to the DLT algorithm (cf. Sections 2.5.1). We can extract one linear equation from every correspondence, with 9 unknowns and the following form

$$\mathbf{x}'^T F\mathbf{x} = 0 \iff vec(\mathbf{x}'^T F\mathbf{x}) = 0 \iff (\mathbf{x}^T \otimes \mathbf{x}'^T)vec(F) = 0.$$

The full system can be written as

$$A\mathbf{f} = \mathbf{0}$$

where A is $n \times 9$.
Because the equations are homogeneous, the solution **f** cannot be unique. It must be $rank(A) \leq 8$, which holds as equality in general. Therefore, the minimum number of correspondences is generally set to 8, and in such case the solution is the right null-space of A, which can be easily computed by solving the linear equations. The rank of A might become 9 in presence of noise, which demands for more correspondences and a

least squares solution.
Once the fundamental matrix has been computed, the camera matrices and scene structure can be evicted. Let us first consider the matrices P and P'. These are commonly found as follows:

$$P = [I_{3\times3}|\mathbf{0}]$$
$$P' = [[\mathbf{e}']_\times F|\mathbf{e}']$$

where P is set as reference. Once both have been fixed, one can express the rays corresponding to image points ($\mathbf{x}$, $\mathbf{x}$') thanks to the projective relation 2.5. The intersection of such rays in $\mathbb{R}^3$ yields the desired 3D point location.

3 | Problem formulation

This chapter provides a formulation of our problem, introducing the notation we will be using throughout the work.

Inputs We are interested in retrieving the pose of a camera starting from a query image and a 3D reconstruction of a scene. Formally, our inputs are:

- the query image $\mathrm{I_q} \in \mathbb{R}^{H \times W \times 3}$, with a related set of keypoint-descriptor pairs $\mathcal{Q} = \{(\boldsymbol{x}, \boldsymbol{f}) \mid \boldsymbol{x} \in \mathbb{R}^2, \boldsymbol{f} \in \mathbb{R}^D\}$
- a set of database images $\mathcal{I} = \{\mathrm{I}_d\}$, from which we may extract the following information through SfM (cf. Section 4.3.1):
  - a 3D reconstruction of the scene $\mathcal{X} = \{(\boldsymbol{X_j}, \mathcal{P}_j, \mathcal{V}_j)\}$, where $\boldsymbol{X_j} \in \mathbb{R}^3$ are the point locations, $\mathcal{P}_j = \{\boldsymbol{d}_1, ..., \boldsymbol{d}_{n_j}\} \subset \mathbb{R}^D$ are two or more associated descriptors and $\mathcal{V}_j$ includes directions and distances of observation of the point at reconstruction time
  - accurate poses of all cameras $\{(\mathrm{R}_d, \mathbf{t}_d)\}$

Output The output is a pose for the query image:

$$(\mathrm{R}, \mathbf{t}), \ \mathrm{R} \in SO(3), \ \mathbf{t} \in \mathbb{R}^3$$

Objective Given a query image q we aim to retrieve its pose $(\mathrm{R}, \mathbf{t})$, minimizing the *position error* and *orientation error* of the camera with respect to its ground truth pose $(R_q, \mathbf{t}_q)$.

The position error is defined as the Euclidean distance between predicted and target camera centre. Note the centre is not the vector $\mathbf{t}$, but it can be retrieved from it as from Equation 2.4 :

$$E_p = \|\mathbf{C} - \mathbf{C}_q\|_2.$$

The orientation error is the minimum angle to re-align the predicted to the target coordinates:

$$E_o = |\alpha|, \;\; 2\, cos(|\alpha|) = \mathrm{tr}(\mathrm{R}_q^{-1}\mathrm{R}) - 1.$$

In literature, it is common practice to evaluate the localization algorithms based on the percentage of queries registered within given thresholds for both errors. Three degrees of accuracy are usually evaluated, with respective thresholds:

- *High precision*: 0.25m, 2°
- *Medium precision*: 0.5m, 5°
- *Coarse precision*: 5m, 10°

Assumptions Because modern cameras have access to a variety of metadata, we may assume additional information is provided for each image we will use, both from the training and query datasets. This comprises camera intrinsics and a rough prior for the location through GPS data. Furthermore, in several applications one can estimate the gravity direction $\boldsymbol{g}$ and height of the shot z_0. These allow further restriction of the number of equations needed to estimate a pose.

We may also assume that a semantic segmentation $\mathrm{M} \in \{1, ..., L\}^{H \times W}$ of all images is available. The segmented classes could be quite high level (e.g. *building, vegetation, vehicle, person, ...*), or more fine-grained. In the latter case, which we discuss in Section 4.6.2, the classes will not correspond to specific semantic classes, but will preserve robustness to variations of appearance.

Challenges We outline here some of the most relevant issues to be dealt with during general localization and in the specific long-term setting:

- *quality of data*: especially in applications where the capturing device is in motion, the resulting images might suffer from blur and saturation, thus hindering the task of describing the image. Increasing uncertainty should be expected in the detections of keypoints, that is the noisy position of a keypoint $\boldsymbol{x} = \boldsymbol{x}_t + \boldsymbol{\eta}$ will see an increase in the magnitude of Σ, with $\boldsymbol{x}_t$ being the "true" keypoint position and $\boldsymbol{\eta} \sim (\mathbf{0},\ \Sigma)$ a random variable with null mean and variance Σ.
- *computational and memory load*: we typically store very large models. For each image thousands of descriptors are stored, and because the scene needs to be accurately triangulated millions of 3D points are usually saved in the SfM reconstruction of a

city-scale scene. Therefore, exhaustive matching procedures like nearest neighbor search appear unsatisfactory to scale to the required model size.

- *structural ambiguities*: large-scale localization also makes it easier to match to visual and geometric structures that are very similar but unrelated. Ambiguities are especially likely to arise in zones rich in vegetation, or urban scenarios with consistent architectural style. Moreover, objects like road signs, vehicles and shops signs can be deceiving, as these appear in different locations too.

- *long-term variations and descriptor variability*: the descriptors used for creating correspondences are built to be invariant and discriminative. Yet, there is a trade-off between such invariance and ability to discriminate between different scenes. In relatively easy localization problems, the queries are visually similar to database images. Therefore, descriptors are able to pass a quite tight Lowe's ratio test, and false correspondences are mostly filtered out at this stage. However, in case of long-term variations, including day-night, seasonal and radical viewpoint variations, appearance may vary drastically. In such cases, descriptors related to the same keypoints will be farther in space. Formally, this means a query descriptor $\boldsymbol{f}_i$ could have similar or even larger distance to its corresponding descriptors $\boldsymbol{d}_j \in \mathcal{P}_i$ than to unrelated descriptors $\boldsymbol{d}_j \in \mathcal{P}_h$, $h \neq i$. Consequently, the probability for matches to be rejected by Lowe's ratio test will increase and localization will suffer from *match scarcity*, which could compromise the ability of RANSAC to find a large enough consensus set and thus decrease the probability of successful localization.

Figure 3.1 reports images which display examples of some of these issues.

(a) Challenging query images from the CMU Seasons dataset. Image and data from [4].

(b) Challenging query images from the Oxford RobotCar Seasons dataset [25]. Image adapted from [37].

Figure 3.1: Query images captured across different conditions. Images in the CMU Seasons dataset exemplify seasonal changes, as well as shadowing, while images in the RobotCar Seasons dataset exhibit blur and day-night illumination changes.

4 | Related work

In this section we outline previous research efforts in visual localization, describing how the state of the art techniques compare to our methods.

4.1. A taxonomy of visual localization methods

The problem of localizing from images can be cast in different versions, according to its objectives and use-cases. For example, only certain applications have GPS information available as a location prior, and when the location is roughly known, it still comes with noise. In some cases, it is necessary to estimate pose and intrinsics of a camera, while other applications assume the latter are known. The database might cover a restricted area of interest (e.g. a building, a university campus), a whole city or the whole world. Many more variations are possible, yet two sub-classes of problems can be identified based on the database scale and required precision of the estimate. These are *visual place recognition* and *camera pose estimation*.
Visual place recognition (VPR), aims at providing a coarse estimation of the location of a shot. Often, the problem is considered to be solved by successfully retrieving the correct place tag. The procedure consists in scanning for the nearest neighbor in a large-scale database of geo-tagged images, and transferring its tag to the query image. In this case, the challenges are represented by performing efficient comparison of images, given the large database size, and discriminating among different places with similar appearance, which are frequently present at such large scale. Note in this image retrieval approach the pose is not directly inferred. When available in the database images, the query pose might be roughly estimated with the nearest neighbour's pose, or by interpolating neighbor poses. Several applications require a much higher precision , while often encompassing smaller-scale scenes. In in camera pose estimation, therefore, an accurate pose is sought, by leveraging denser and more accurately annotated datasets.

Pose estimation techniques are traditionally based on geometric reasoning. They can be divided into *image-based* and *structure-based* methods. For the latter, 2D-3D matches are used to infer the pose as explained in Section 2.5, while in the former case some strategies

use 2D-2D matches between query and database images to solve for the absolute pose. For example, [50] match rectangular structures in pairs of views, and use such matches to deduce the pair (R, $\mathbf{t}$).
More recently, learning-based pose regression techniques have emerged. These methods receive as input the images and train a neural network to directly regress a pose.

The focus of this work is camera pose estimation. We will adopt the traditional approach, and employ structure-based techniques, as these still represtent the state of the art for most localization benchmarks.

4.2. Image description

Although visual localization problems come with different goals and constraints, they share the need to effectively encode images' appearance. Particularly, finding image descriptors is often the starting point for both techniques. Image contents can be observed at local level or at global level. The former focuses on the information carried by a small regions of the image. For example, these could be the colour of a pixel, its semantics, or the color gradients of the pixel with respect to a few of its neighbors. For this type of description small patches of pixels are analyzed independently of other patches and regardless of the context of the image. In this sense, local descriptions are not unique, since the same image patches might appear in different images in very different positions, scale and orientation. Only when local features are gathered and analyzed in group the peculiarity of one image emerges from another. Conversely, a global description naturally aims at describing the picture on the whole, and is thus a discriminative property of the image it represents. Although giving a comprehensive summary of the image content, a global descriptor carries no information on local features, i.e. no details on the position, scale and orientation of the features are kept.

Both local and global description techniques are widely used in the context of localization tasks, since they provide useful information to recover similarities in a wide pool of candidate database images.
Global descriptors are by nature compact and well suited to computing pairwise similarity between images. Indeed, provided that a distance is defined on the space of descriptors, when every image is described by one global vector comparison is immediate. Conversely, using local features, entails the comparison of all pairs of features from both images, dramatically increasing the number of operations to be done.
Local description is more robust than global description. Indeed, even when a significant part of the image content is occluded or has changed appearance due to long-term

visual variations (e.g. seasonal changes, changes in illumination), the presence of a few local matches and their position with respect to one another might still make localization possible. Furthermore, the ability to estimate the pose of an image is conditional to having information on several matching points, in order to triangulate the accurate camera position and orientation. In summary, from case to case there is a need for both representations, and sometimes both are used (cf. Section 4.6.1).
We now explore a few successful description techniques for both local and global descriptors, and anticipate their usefulness within the problem of camera pose estimation.

Scale Invariant Feature Transform (SIFT)

The *Scale Invariant Feature Transform* descriptor, first proposed by [24], is a technique to obtain local image descriptors with desired properties of invariance to changes of scale and rotation of the feature, as well as some invariance to illumination conditions and viewpoint. Such properties are particularly interesting for the repeatability of detection and description in different images. Description follows a sequence of steps:

1. An efficient search is performed across the image at different scales to identify candidate interest points. Identification is carried out through convolution of the image with a difference of Gaussians, as an approximation of the laplacian of Gaussians. In particular, maxima of the so-obtained curve are selected. This is a standard technique that aims at pointing out the locations and scales of edges in the pictures, that is regions in the image where the intensity of pixels suddenly changes.

2. Starting from promising locations, keypoints are chosen within the surrounding regions based on their stability. This is achieved by interpolating a 3D quadratic function to the difference of Gaussians computed at the previous step, and acts as a regularizer. Keypoints are further filtered by checking the surrounding area has a sufficient contrast around the edge.

3. The described region is assigned one or more orientations, based on the directions with the strongest appearance variation. The gradient of pixel intensity is again used to search for peaks in discretized directions. The highest peak is selected, along with all other local peaks with comparable intensity. Thus, multiple keypoints can originate from one described region. This guarantees stability in presence of rotations, since the final description is given relative to the orientation.

4. Finally, a descriptor is assigned to the selected region. To maximize invariance to viewpoint changes the proposed approach computes a histogram of the image gradients in the described region. The histogram is filled by uniformly sampling

gradients in the region, and accumulating votes for discretized orientations based on the gradient magnitude, with a Gaussian weight falling lower as distance from the keypoint increases. The resulting histograms are 128-dimensional, since 8 discrete bins are used to describe every subregion from a 4 x 4 grid over the whole described region. These are further normalized to contrast variations in illumination.

Overall, the histogram of gradients representation demonstrates robustness to local appearance variations, and is thus well suited to the purpose of matching keypoints in different images.

The authors originally proposed to compare the SIFT descriptors by Euclidean distance, and validated the effectiveness of such embedding for matching even in presence of clutter and occlusion. Conversely, later research by [2] proposed to compare descriptors in a more natural space, namely that of probability distributions over the 128-dimensional space of discrete orientations. The descriptors are thus l^1 normalized and compared with the Hellinger distance, $d_H^2(\mathbf{x}, \mathbf{y}) = 1 - \sum_{i=1}^{128} \sqrt{x_i\, y_i}$.
Another interesting modification of SIFT that was proposed specifically for large-scale and long-term place recognition is the DenseSIFT [47]. The challenging scenario of day-night variations is shown to completely disrupt not only the descriptor, but also the keypoint detectability. Indeed, when appearance changes, very different keypoints are found using the classical difference of Gaussian detector. The proposed solution is then to sample keypoints on a predefined dense grid of locations on the image, and describing constant size patches around them with RootSIFT. This approach is naturally suited to views from approximately the same viewpoint. Hence the authors combine the DenseSIFT with additional synthetic views of the scene, generated through Google maps data, including panoramas and depth maps. In our setting, however, we do not have available depth maps of the scene, and we will rather use sparse descriptors.

SuperPoint

Inspired by the wide success of convolutional neural networks (CNNs) in many relevant tasks in the image domain, [11] propose a competitive approach as alternative to the hand-crafted SIFT descriptor.
The SuperPoint approach well represents the challenges of learning a suitable keypoint representation, arising from the fact that the concept itself of an interest point is ill-defined. Indeed, we require from a keypoint that it is as repeatable and as stable as possible across different viewpoints, illumination conditions and orientations of an object. In reality, not all objects have obvious endpoints and, even if they do, these are not necessarily visible from all directions and under all lighting conditions. Thus, we are first

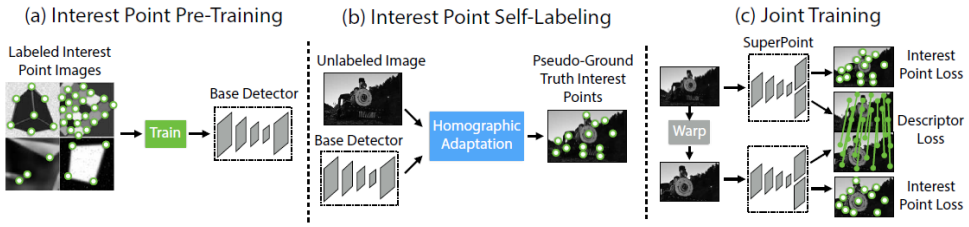


Figure 4.1: The three phases of SuperPoint training. (a) A base detector architecture is trained on synthetic images representing geometric shapes, whose interest points are unambiguously detectable. (b) The base detector is trained with homographic adaptation to generate pseudo-ground truth keypoints of available dataset images. (c) The final architecture is trained to output the same keypoints and descriptors for corresponding points in the image. Figure from [11].

challenged by labelling interest points, and subsequently detecting them. SuperPoint's derives both keypoint locations and description from direct training on the dataset, ensuring the best detection for the goal of repeatability.

We now briefly explain the self-supervised training mechanism proposed in the paper, and derive some key ideas on the capabilities of self-supervised learning in the context of description and matching.
As illustrated in Figure 4.1, SuperPoint training is composed of three phases.

Firstly, the keypoint detection task is cast in a domain where labels can be easily obtained. For this purpose, synthetic geometric shapes are created, and keypoint labels are automatically assigned to angles, centres of ellipses and segment endpoints. A classification network is used to predict the probability of a small region in the image with size 8 x 8 pixels to contain a keypoint, and in that case the pixel at which the potential keypoint is found. Thus, the problem is cast as a 65-class classification, where the 65^{th} class corresponds to no interest point, and the others all represent one pixel in the region. After training in this first phase, the *base network* is able to detect a wide range of interest points in the synthetic domain, but still falls behind of traditional methods on real world images. Thus, a second phase of training is used as a domain adaptation step. The proposed procedure is named *homographic adaptation*, and consists in refining the predictions of the base network on a database image by averaging detected keypoints for a number of random homographies of the same image. The so-obtained ground-truths are now stable to a wide variety of transformations that are chosen to be as representative as possible of real world scene transformation, and can therefore be used for a final joint training of the *SuperPoint network* to output keypoints and their descriptors. In particular, the D-dimensional descriptors are trained in a metric learning fashion, promoting descriptors

similarity where these represent the same keypoint, and encouraging their separation for different points.

The experimental results show a dominance of SuperPoint on other traditional descriptors such as SIFT. This type of descriptor is employed in some current state-of-the-art localization pipelines, such as HF-Net (cf. Section 4.6.1). However, as other learning based methods, a drawback of the SuperPoint approach is that the network might not generalize to unseen conditions. For this reason, the SIFT descriptor and its variations are still widely used in literature, which motivates us to also adopt this type of traditional representation.

Global descriptors

Global descriptors are particularly fit for retrieval tasks, for their lightweight nature and ease of comparison. In such situation, one can attempt to make descriptors very "general", so that they work well on images regardless of their domain, or tie them to the specific dataset by learning the best representation on that domain. The latter methods employ CNNs trained with contrastive losses, which pull together positive samples and push apart negative ones. The former methods, instead, are often built on local representations of the image, so that despite the loss of local information when aggregating descriptors, the whole image content weighs in the description.

Starting from the former methods, a first approach is called *Bag-of-Visual-Words* [40] and its idea is to describe the image content through a fixed "vocabulary", a partition of the space of local descriptors usually obtained with K-means clustering on the union of all local descriptors from the database. The whole image is then represented as histogram of the occurrence of each word, and intuitively images with similar content will carry similar global descriptions. However, this approach is quite simplistic, since visual words can include quite a variety of descriptors, and we might still want to discriminate among them.

A more fine grained description can be obtained by storing residuals within each visual word, rather than simply the count of descriptors falling in that word. This approach goes under the name of VLAD [17]. The VLAD representation is naturally more fine grained than the Bag-of-Visual-Words one: if the local descriptor space is D-dimensional and we choose a quantization of W words, the global descriptor is a WD dimensional vector, whose components from wD to $(w+1)D-1$ are given by the sum of residuals of

descriptors falling in the w-th visual word from the centre of the word, that is

$$\mathbf{d}^G(I) = \begin{bmatrix} \sum_{i=1}^{N_1} \mathbf{d}_i^L - \mathbf{c}_1 \\ \dots \\ \sum_{i=1}^{N_K} \mathbf{d}_i^L - \mathbf{c}_K \end{bmatrix}.$$

The comparison of VLAD descriptors is usually performed by dot product. Therefore, images which have opposite residuals contribute negatively to the measure of similarity. As in the case of local descriptors, hand-crafted methods- for global description can be greatly improved by deep learning. However, as mentioned above, the risk of relying on deep learning is to fail generalizing to broader datasets rather than the training one. For this reason, it has been proposed to maintain a structure similar to VLAD but with learned parameters. In practice, the procedure to construct the VLAD descriptor is not differentiable due to the assignment of each local descriptor to a visual word, which makes it impossible to directly train a VLAD descriptor. Hence, [3] propose a layer with a soft version of cluster assignment and design a full architecture to train *NetVLAD* descriptors. Training on a contrastive loss with positive and negative examples, they are able to exceed the performances of hand-crafted descriptors on datasets not seen at training time, confirming the ability to generalize of the method.

4.3. Key steps of localization

As introduced in chapter 1, the localization pipeline is composed of several steps, summarized in Figure 4.2.

Before localizing, the available dataset has to be processed to obtain a 3D reconstruction of the scene. After reconstruction, keypoints and descriptors are extracted from the query image, usually with local features such as SIFT descriptors [24] (cf. 4.2). The reconstructed point cloud is also enriched with local features from the images it was reconstructed from, so to enable matching.
Localization proceeds with matching the query descriptors to database descriptors, filtering the resulting correspondences and estimating a pose with a pose solver, inside a robust estimation framework such as RANSAC [13] to prevent outliers from disrupting the model.

We now see these phases in greater detail.

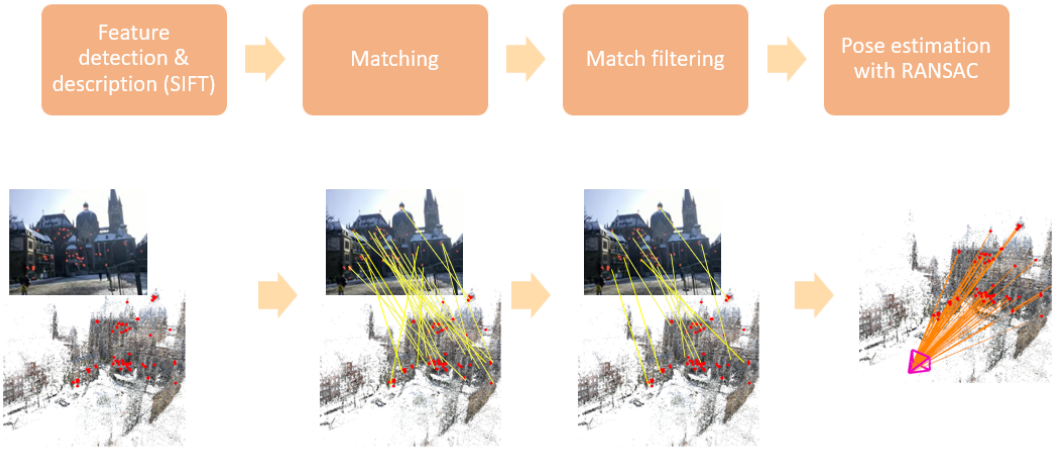


Figure 4.2: Standard structure of an online visual localization pipeline. Having previously reconstructed the scene, the query image is described, then matched to the point cloud. Before pose estimation, matches are filtered to discard mismatches. The so obtained correspondences are then used for pose estimation with a pose solver within a RANSAC framework. Figure adapted from [32].

4.3.1. 3D reconstruction from database

The first step consists in reconstructing the database scene offline. A popular tool for this is incremental Structure from Motion (SfM).
The standard pipeline for SfM starts with identifying overlapping images and their relative pose. To this end, images are matched with 2D-2D correspondences between SIFT descriptors [24], which are subsequently tested for distinctiveness. Geometric verification with RANSAC further discards outliers and outputs a relative pose between the two images. Once the relationships among all pairs of images have been summarized in a *scene graph*, with images as nodes and verified overlaps as edges, incremental reconstruction of the scene is performed following the graph edges, and absolute poses of all images are found. To initiate the process, two initial images have to be carefully selected to generate the first partial reconstruction of the scene via triangulation. Subsequently, a new image from the neighbor nodes of the starting images can be registered (an absolute pose is estimated) and additional points not yet in the 3D point cloud can be added if they appear in at least two images from the previously visited ones. Bundle adjustment, a procedure for readjusting the estimated absolute poses and 3D point positions by minimizing reprojection errors, is also needed after triangulation and pose estimation to ensure the reconstruction does not accumulate errors.

Throughout this work, we will assume 3D SfM models are reconstructed with the pipeline of [39]. Such work is based on the standard pipeline, with additional improvements to enhance the reconstruction effectiveness while limiting the computational effort.

4.3.2. Match formation and filtering

The matching phase takes as inputs collections of keypoints and descriptors for the query image and the database. Let $\mathcal{Q}, \mathcal{F}$ be the sets of keypoints and descriptors for the query image, and $\mathcal{X}, \mathcal{P}$ points in the point cloud and the associated descriptors. Note that every point could have multiple associated descriptors.
The goal of matching is to find a subset $\mathcal{M} \subset \mathcal{Q} \times \mathcal{X}$ representing putative matches to be tested in RANSAC. It is clearly prohibitive to investigate all possible matches, since the average number of descriptors in a query is over 10^3, and for the point cloud it is commonly in the order of $10^5 - 10^6$. Testing all pairs would be impractical for the sampling scheme of RANSAC.
It is worth observing that there can be no more than $|\mathcal{Q}|$ correct matches, but there are usually fewer. This is either due to backgound noise in the keypoint detection phase, the disappearance of the keypoint from view or the disappearance of the interest point in the real 3D scene. In fact, when localizing with a long time span, the scene is not static.
The SIFT descriptor [24] is designed to guarantee some invariance to visual variations, and ideally to be distinctive, that is close to similar keypoints and far from others. Similarly, CNN-based descriptors such as SuperPoint [11] learn an embedding with metric learning, so that by construction corresponding points lie close to each other, and far from unrelated points. Hence, standard matching schemes establish corresponding points by finding the nearest neighbor in descriptor space.

Once correspondences have been formed, a pruning step usually follows to discard matches that are most likely outliers. The authors of the SIFT descriptor themselves propose a widely used strategy, the ratio test [24], specifically designed to single out matches that were formed by chance. The test relies on the distinctiveness of the descriptors, and only accepts a match if there is a sufficient gap between the distances d_1, d_2 of the source descriptor with its two nearest neighbors from the set of target descriptors to match to. That is, we will only retain a match if

$$\frac{\|d_1\|}{\|d_2\|} < threshold \tag{4.1}$$

where the threshold is commonly fixed between 0.7 and 0.9 depending on whether a more or less aggressive filtering is preferred.

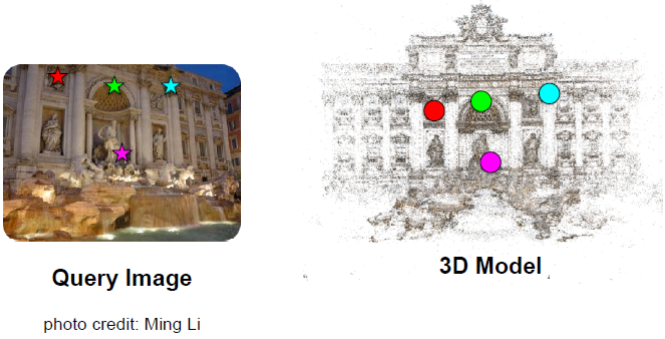
It is interesting to reflect on the direction of matching. Indeed, one could assign to every image descriptor its nearest point cloud descriptor (2D-3D matching), assign an image descriptor to the point cloud descriptors and retain the top match for every query descriptor (3D-2D matching), or check that 2D and 3D descriptors are mutually nearest neighbors. A few studies were made to compare the 2D-3D versus 3D-2D matching approaches, reaching conflicting conclusions.

The work of [23] argues the most convenient direction of search is from 3D to 2D, namely *from Point to Feature* (P2F), since exploiting the richer information of the dataset can help reduce the computation overhead of nearest neighbor search for the whole point cloud. For instance, knowing how often a point is observed in the dataset naturally yields a probability distribution over the 3D descriptors for prioritizing the search. Moreover, once valid matches have been found, it is likely to find more in the nearby areas, which also restricts the number of matches to be searched. Thus, they assign to each point a *visibility* score based on the number of times it was observed in the database. The 3D-2D search starts from a selection of the most visible points, called *seed points* from different areas in the model. Whenever a match is found, the nearby points receive a higher priority, and are visited soon. The search is terminated upon finding enough correspondences. Given that only few mismatches will be accepted by chance (one in 500 on average, as reported by the authors), the prioritization scheme should ensure the algorithm can quickly terminate once it finds the right location in the 3D structure.

Conversely, [36] report the superiority of the direct approach (2D-3D) both in computational terms and for the discriminative quality of the ratio test. Regarding the first observation, they note that the search has a cost of $\mathcal{O}(|\mathcal{F}|\ log|\mathcal{P}|)$. This is extremely competitive with respect to the opposite search since usually there are far more points in the point cloud than in the query. As for the second observation, they report that the ratio test is more effective when applied to prune ambiguities in the 3D domain, which are more frequent and thus tend to generate more outliers. For these reasons, the 2D-3D approach is usually preferred as first instance for the search, although a more refined approach can also perform *back matching* on some selected points (cf. Section 4.3.2). Yet, the direct matching approach remains too computationally expensive even in the 2D-3D case for most applications, which demands for a clever pruning of the query descriptors to search from. Indeed, [36] note that fewer than 10% of matches has a corresponding point in the point cloud. We now explore some solutions to this issue.

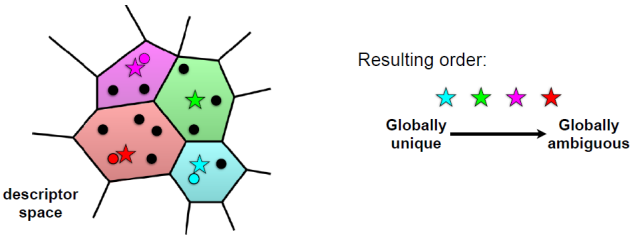
Vocabulary-based priority search (VPS)

With the aim of selecting the most likely matchable points, [33, 36] propose to quantize the space of descriptors in a Bag of Words fashion (cf. Section 4.2) and use the size of resulting visual words as a priority scheme for the search.



(a) Query image and keypoints

(b) 3D reconstruction of the scene and corresponding features



(c) Quantization and prioritization

Figure 4.3: Example of prioritization adopted in the VPS approach. All images are from [32].

First the query features are assigned to their visual words by finding the nearest centre. The assignments are then sorted from smallest to largest visual word (in terms of number of database descriptors) and an attempt to match the query features is made by searching the closest neighbor among database features within the visual word, with a subsequent ratio test. The search stops when a sufficient number of descriptors is found, which makes the prioritization useful for a quicker search. Such number is experimentally tuned at 100 correspondences.

Consider, for instance, Figure 4.3. The stars correspond to features in the query image, while the points are database features. The light blue feature is preferred to the red one, as there are fewer descriptors in its search space, and it is more likely that the ratio test will not reject the correspondence. This is also a speed-up in the nearest neighbor search, as fewer descriptors have to be searched to find the nearest neighbor. However,

the procedure might bring inaccuracies due to quantization errors, namely the assignment of true neighbors to different words with subsequent loss of the correspondence.

Active search

To overcome the quantization artifacts of the previous method, [34] have observed that a hierarchical quantization would often allow to recover lost matches. The proposed way to exploit this fact without undermining the computational advantage of VPS is by triggering a 3D-2D search on a coarser level in the surroundings of a verified match. The method is known as *active search*(AS).

The algorithm of active search naturally integrates to the VPS pipeline. The rationale for the method is that whenever finding a valid match, there will likely be more in the surroundings. Therefore, a fixed neighborhood of N_{3D} 3D points could be prioritized for an inverse search –namely in the 3D-2D direction – to speed-up the matching, as well as recovering matches lost due to quantization and structural ambiguities across different locations of the 3D model.
The prioritization scheme of VPS is easily adapted to this strategy, and the most important design choice is whether to add the 3D-2D search immediately after verifying a match (*direct prioritization*) or by proceeding according to the cost of search, just as in the 2D-3D prioritization choice (*combined prioritization*). The search is concluded after finding N correspondences, just as in VPS. Once the matches are formed, a pose is estimated with a standard pose solver inside a RANSAC scheme.

These approaches are particularly efficient and thus widely used. However, they have been shown [37] to perform poorly on challenging long-term scenarios, such as those rich in vegetation. In those situations, it is not advisable to terminate the search of matches early as these methods do, since a high probability of mismatches combined with overall match scarcity may cause failure of pose estimation.

4.3.3. Pose estimation and verification

Having discussed the creation and filtering of matches, we explore how these are used to infer the camera pose via geometric pose solving within a robust fitting framework.
Let us specialize the methods introduced in the domain background (cf. Section 2.5.2, 2.6) to the pose estimation problem.
The standard fitting technique is the P3P algorithm used within the RANSAC framework, as described in the work of [13]. 3 point correspondences are required as minimal sample size to find a model (in case the configuration offers multiple solutions, all can be tested

looking for the largest consensus), and subsequently consensus is evaluated by counting how many projected points fall within error ellipses of the matching points in the image. Such ellipses depend on how much the fitted model is sensitive to fixed perturbation in the three image points.
Since a minimal sample size of 3 is used, we can compute with equation 2.6 a reasonable number of iterations for the algorithm. For example, if we choose $\alpha = 0.99$, an outlier probability of 0.95% has $N_{iter} \approx 37000$, while if the probability grows to 0.97%, iterations increase to $N_{iter} \approx 170600$.

Reducing minimal sample set size

The major drawback of RANSAC is the high computational cost of fitting and evaluating thousands of models. Since the number of models to try to reach a sufficient confidence on the chosen model has super-exponential growth in the minimum sample size, several efforts have been directed towards reducing this size. It is possible, for example, to reduce the number of required correspondences by introducing constraints on the pose.
One notable example is when the gravity direction is known in camera coordinates, and can be solved with 2 points correspondences as proven in [21]. Knowing the gravity direction means removing two degrees of freedom in the 6-DoF pose of calibrated cameras, specifically from the rotation matrix.
Intuitively, if the gravity direction is known, there is a possibility to measure the angle between the known vertical direction and two of the axes of the camera reference system – the third axis follows from the other two – which only leaves ambiguity of a 1-dimensional rotation around the vertical direction. In practice, the authors derive two independent polynomial equations of degree two for every point correspondence, thus requiring exactly two correspondences to compute a pose.

Designing a strategy for sampling models

An alternative strategy to improve the speed in RANSAC, presented in [8], builds on the observation that RANSAC sampling strategy is "blind" to any prioritization in the dataset, since samples are selected randomly. However, the data generation process often offers a proxy of the quality $q(\cdot)$ of data, which can be used to speed-up the fitting. In the case of absolute pose estimation, the ratio between descriptor distance with first and second nearest neighbor has been considered as prioritization strategy, under the assumption that less ambiguous matches are more likely to be correct. The authors propose a *Progressive Sample Consensus* (PROSAC) algorithm.
PROSAC is designed to yield not-worse-than-RANSAC performances in the worst case,

when data do not have a specific ordering and are independent of the quality function $q(\cdot)$, while terminating much earlier in favorable cases. To achieve this, correspondences are sampled from small, high quality sets first, and their size is gradually increased to include the next correspondence in the order dictated by match quality, until the termination condition is reached.
Before sampling, correspondences have to be sorted in descending order with respect to the quality function, namely with respect to the following relation:

$$i < j \implies q(\mathbf{m}_i) \geq q(\mathbf{m}_j)$$

for every two correspondences $\mathbf{m}_i, \mathbf{m}_j$. We denote the resulting sequence with $\mathbf{m}_{(1)}, ..., \mathbf{m}_{(N)}$. Consider sets $\mathcal{S}_n = \{\mathbf{m}_{(1)}, ..., \mathbf{m}_{(n)}\}$. One crucial assumption here is monotonicity with respect to $q(\cdot)$ of the probability of a match to be an outlier p, i.e.

$$q(\mathbf{m}_i) \geq q(\mathbf{m}_j) \implies p(\mathbf{m}_i) \leq p(\mathbf{m}_j).$$

Under this assumption, sampling from the set $\mathcal{S}_n$ maximizes the probability to pick good matches on a subset of data of size n. In the worst case, when probabilities are all equal, the sampling strategy will not be worse than another random strategy.
The algorithm starts with the smallest sampling set $\mathcal{S}_m$ and alternates sampling from the current set and adding one more correspondence to the set, until termination.
To define how many samples should be extracted from each set $\mathcal{S}_n$, the worst-case scenario is considered. In fact, to achieve RANSAC-like performances when the quality function is uninformative, the amount of sampled matches of each increasing set should be equal to the average number of samples belonging to sets of the same dimension when sampled from the largest set $\mathcal{S}_N$.
This amount is computed through a *growth function*. To find the growth function, we first set a fixed number of total samples T_N. The authors propose $T_N = 200000$. The average number of samples within these T_N containing points from $\mathcal{S}_n$ only is then

$$T_n = \frac{\binom{n}{m}}{\binom{N}{m}} T_N$$

with $m = 3$ the sample size. From this relation, simple computations show the recursive relation between T_{n+1} and T_n

$$T_{n+1} = \frac{n+1}{n+1-m} T_n.$$

The sequence $\{T'_1, ..., T'_N\}$ is obtained from $T'_{n+1} = T'_n + \lceil T_{n+1} - T_n \rceil$, $T'_m = 1$, and finally the growth function can be defined as

$$g(k) = min\{n : T'_n > k\}.$$

In the comparison with RANSAC, PROSAC proves up to 100 times faster than RANSAC, making it a preferable strategy.

The idea of sampling with prioritization is also at the heart of the semantic consistency method of [44]. In their work, the quality of every match is used as a bias for sampling points. This procedure relies on the validity of the quality function even more than PROSAC and is a useful tool to accelerate the correct model among very large collections of matches.

4.4. Large-scale visual localization

The localization pipelines outlined above are very effective and efficient for relatively small models, where ambiguities are easy to filter out. However, as the model size increases, especially in urban scenarios, it is more and more likely to encounter similar structures, which in the best case make it difficult to match individual features and in the worst case can cause localization to fail completely.
To tackle this challenging situation, literature has focused on exploring an increasing number of matches, up to multiple matching hypotheses, in an efficient way. We now explore three such methods in detail.

4.4.1. Matching hyperpoints

Aiming at urban settings with a large amount of repeated structures, the authors of [35] propose to go beyond traditional match filtering with a strategy based on three key points:

1. An extremely fine visual vocabulary is created to store all point descriptors in the database.

2. Lowe's ratio test is not performed to validate one-to-one matches between query

and database features. Instead, several matching database descriptors from the same word as the query descriptor are kept (one-to-many matching strategy). The collection of these descriptors is called a *hyperpoint*.

3. *Consistent* hypotheses are formed out of the possible configurations dictated by hyperpoints. Each such hypothesis votes for an image in the database, and based on the largest amount of votes a few final inlier sets are selected for robust pose estimation.

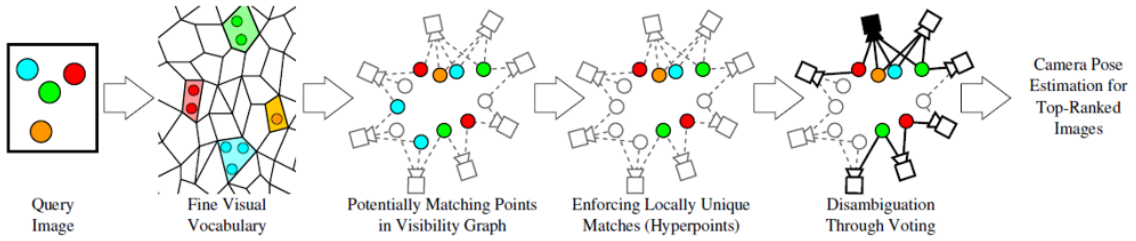


Figure 4.4: An illustration of the matching procedure with hyperpoints. At first, descriptors are extracted from the query image, and assigned to visual words from a fine visual vocabulary. All database descriptors that belong in the same words as the query descriptors are potential matching points for that feature. Hyperpoints are subsequently formed by a subset of such points, by enforcing no co-visibility among the remaining points. For example, we can see two light blue points were removed since they were observed by the same camera. Finally, cameras are ranked in terms of observed matches in the hyperpoints, and a pose is estimated starting from the top ranked camera (highlighted in black in the picture). Figure from [35].

Let us now define the data representation and subsequently examine the detailed strategy, whose steps are illustrated in Figure 4.4.
The inputs are the query keypoints and descriptors, for example classic 128-dimensional SIFT keypoint descriptors, and a bipartite graph encoding 3D structure and visibility information of the database. With a slight abuse of notation, we will refer to image keypoints and descriptors simply as *features* in the feature set $\mathcal{F} \subset \mathbb{R}^D$, and we will address 3D points and their descriptors simply as *points*, whose set is $\mathcal{P} \subset \mathbb{R}^3$. The resulting visibility graph, which connects 3D points $X \in \mathcal{P}$ and database images $c \in \mathcal{C}$, has edges in the set E for every pair of point-camera such that the point was detected in the camera. The graph is then $\mathcal{G} = (\mathcal{P} \cup \mathcal{C}, E)$.
The goal is to find a set of matches $\mathcal{M} = \{\{f, p\} \mid f \in \mathcal{F}, p \in \mathcal{P}\}$ that yields a good pose estimate in presence of large-scale ambiguities.

Firstly, the fine vocabulary structure is chosen to compress descriptor information while keeping discriminative ability. Compared to the 100k typical vocabulary size in [34], the 16M word vocabulary naturally captures much more fine-grained distinctions among descriptors, which make it superfluous to apply Lowe's ratio test on a second neighbor, since either the second neighbor is extremely close to the first – in which case the match would likely be rejected – or the second neighbor is assigned to a different visual word and Lowe's test is not performed at all. In light of this fact, it is of paramount importance to design a strategy to choose among possible ambiguous matches from the same visual word.

The authors observe that, although at local level it may not be clear which descriptor is best to choose, uncertainty is easily resolved at larger scale, namely when considering matches in respect to each other. Indeed, it is unlikely to find the very same collections of matches with similar geometric layout in unrelated places, therefore a voting procedure will let the "right" location emerge.

To better explain the concept let us consider the situation depicted in Figure 4.4. The red 2D keypoint falls in a visual word including two 3D points far away from each other. Their descriptors are quite similar, so that a strict ratio test would likely reject the assignment between the query and 3D red points. However, if an oracle provided information about the rough location of the picture – say we were told to restrict our search around the camera highlighted in black, an information which in practice is provided by a voting-based image retrieval mechanism – there would be no ambiguity on the right match to be chosen. The red dot in the upper part of the picture would be the unique match available, having discarded the one at the bottom. Vice versa, when considering the light blue feature and its corresponding dots, we can observe there is ambiguity even at larger scale. Imagine the right location was in the bottom part of the picture, where two light blue features appear next to each other –for example, this might be the case of a repeated structure. In such situation, knowing about other valid matches would not help choose among the two putative matches, and therefore we prefer to discard the uncertain matches all together.

In other words, it is important to evaluate mutually exclusive hypotheses. Indeed, only those matches which give a unique correspondence when limited to a smaller region of the 3D model are informative. These suitable matches are then defined as *locally unique*. The novelty with respect to the previous methods is that the match needs not be *globally unique*, allowing for much more information to be evaluated for pose estimation. In practice, authors link the definition of locally unique match (f, p) to the property of not having any other putative match (f, p') with p' co-visible with p. A point p is co-visible with p' if there exists a camera c such that both $\{p, c\}$ and $\{p', c\}$ are in the visibility

graph $\mathcal{G}$.
All locally unique points matched to a feature f are gathered in a hyperpoint $H(f)$. Considering the collection of matches $\{(f, H(f))\}$, we would then wish to generate all consistent hypotheses extracting at most one point from every hyperpoint, and rank these hypotheses so to obtain a final inlier set to estimate a pose. This problem is equivalent to retrieving the database image c which observes most points belonging to some hyperpoint. It can efficiently be found by letting every point in all hyperpoints vote for a camera and simply choosing the camera with highest vote count.
Finally, as it is customary in image retrieval, top-N retrieved images are evaluated with RANSAC (whose sampling set may be enriched with further matches found on nearby database images). The final estimate is selected by re-ranking these N images, based on their effective inlier count – a measure of both the number of inliers and how uniformly these are distributed across the image, so to avoid small clusters of very close matches to disrupt the solution.

4.4.2. Localization by scene registration

Another tractable strategy to handle ambiguities [42], not involving image retrieval, proposes to relax the ratio test [24]. The ratio threshold used by the original paper is quite low (0.7), hence filtering a large portion of ambiguities. When increasing the threshold, more valid matches are recovered, but many more wrong matches are also kept. The focus of the paper is thus to devise a pose estimation technique which can explore the space of models faster than RANSAC with very high percentage of outliers. The authors propose an efficient framework based on registration of the query camera to the 3D scene. Instead of estimating a pose through sample consensus, the registration approach consists in finding a transformation of the camera coordinates that aligns the camera to the 3D structure.
All keypoints in an image come with some noise, for example due to imprecise keypoint detection. That is why, when computing the consensus of a certain model, it is not required that 3D points project exactly on the image point, but they should be found in a small region around it, which we will call R_i.
The perspective can be switched from counting inliers in the 2D image plane, to counting them in the 3D space. In this case, the error region $\tilde{R}_i$ is a cone, called *error cone*, with vertex on the camera centre and whose section by the image plane is the region R_i. Inliers are 3D points X_i falling in the error cone $\tilde{R}_i$ of their matched 2D point.
With this novel point of view, the core idea put forward by the authors is to maximize in the camera pose the number of 3D inliers falling into their respective error cones.

Figure 4.5 depicts a sketch of the registration procedure. In the figure, the camera pose initialization is not optimal, as some points are not registered within their cones. Therefore, a roto-translation is applied to the reference system to maximize the number of points in the cones, and the optimal transformation yields the pose estimate. Note that in the picture the camera reference has been kept constant and the 3D structure was moved.

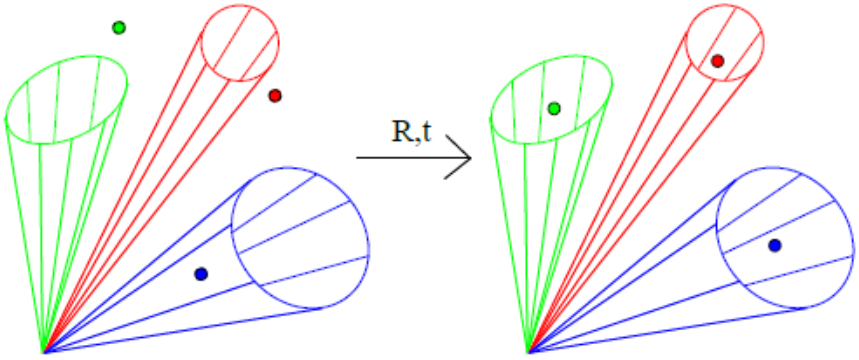


Figure 4.5: An illustration of the registration procedure of [42]. On the left, a wrong pose causes most inliers to fall outside of the error cones determined by the pose. However, by optimizing the relative positioning of the point cloud and camera, a larger number of points can be registered in the cones, as it is seen on the right. Figure from [42].

Given the complexity of this problem, some crucial assumptions are introduced. Firstly, it is required that gravity direction is known in camera coordinates. Secondly, a rough prior on the camera height is needed.
When this information is given, the pose to be estimated becomes simpler: thanks to the known gravity direction, and without loss of generality, we can assume the camera y axis is registered to the vertical direction. If the camera centre is fixed, the pose will be specified up to the camera rotation on a horizontal plane, described by a 2 x 2 rotation matrix. The translation furthermore adds three unknowns, yet with the constraint that the height should only be sought in an interval $[\, h_{min}, h_{max}]$.
In formulas, we are required to find

$$R = \begin{bmatrix} \tilde{R}_{2\times 2} & 0 \\ 0 & 1 \end{bmatrix}, \; \mathbf{t} = \begin{pmatrix} t_x \\ t_y \\ h \end{pmatrix}$$

with $h \in [\, h_{min}, h_{max}]$.

We now imagine moving the 3D point cloud instead of the camera, of course with the same degrees of freedom as the previous situation. With an appropriate change of reference system, it is possible to prove the two movements are equivalent. Although the optimization procedure is based on the theory of KKT points (cf. [12]), which offer a framework for finding $argmax_{\tilde{R},\mathbf{t}}|\mathcal{I}|$, $\mathcal{I}$ being the set of inliers, an interesting further pre-processing step is used to reduce the number of correspondences to search from. Such procedure is inspiring for further methods which we will explore in the following sections. We briefly describe it here.
The algorithm, denominated *Fast Outlier Rejection*, aims at discarding all correspondences which would constrain the model to have too few inliers. The problem is again cast as a registration task, although in this filtering step the geometry constraints can be relaxed to speed up the search of obvious outliers.

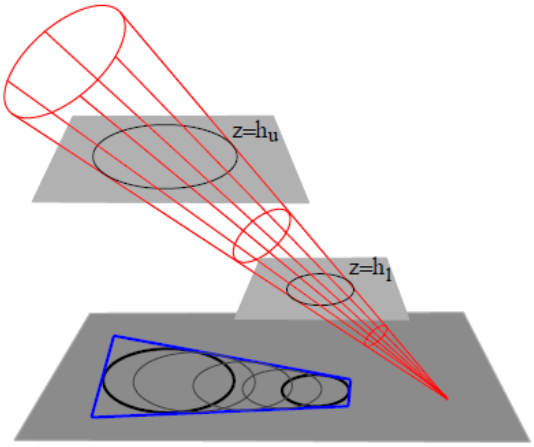


Figure 4.6: The error cone of a 2D keypoint intersected by the camera height boundaries can be projected onto the ground plane. A subsequent approximation to a quadrilateral region makes it easy to verify the inclusion of 3D point projections within the approximate area. Figure from [42].

The approximated registration procedure, also illustrated in Figure 4.6, considers projections onto the ground plane of the error cones and their respective points. Indeed, let us assume as before the camera pose is fixed and the 3D point cloud needs to be registered to the camera. Each error cone corresponding to a 2D point will intersect the planes $z = h_{min}$ and $z = h_{max}$ in a conic section (ignoring the degenerate cases). Overall, projecting this region onto the ground plane yields a composite figure, which can be well approximated by a quadrilateral Q. Thus, we may project also 3D points onto the ground

and transform the problem into an easier 2D-2D registration procedure.
The final outlier filtering is carried out by considering the correspondences one by one, and, in the hypothesis that the chosen correspondence is an inlier for the relaxed model, quickly assessing the maximum number of other inliers – namely points that also fall in their respective quadrilaterals Q_i. If such number is smaller than a threshold, the correspondence may be safely discarded.
The computational complexity of the algorithm is driven by the outlier removal step. The cost of performing a check on one match is found to be $\mathcal{O}(n\,log\,n)$, thereby giving a total cost of $O(n^2\,log\,n)$. The subsequent optimization step goes as $O(n^4_{filtered})$, but since the number of filtered points is much lower than n the impact of the latter slower computation is marginal.

4.4.3. Camera pose voting

Starting from the geometric setup of the city-scale localization procedure in the previous paragraph, and borrowing the same assumptions of knowing gravity direction and a boundary for camera height, the authors of [49] propose an even faster pose estimation algorithm, allowing to handle 1-to-many matches.
While in the previous approach the algorithm had a complexity $\mathcal{O}(n^2\,log\,n)$, with n the number of matches, the voting approach allows to achieve a linear runtime in n, at cost of more memory to store votes for every camera pose hypothesis.
As seen in the city-scale localization paper (cf. Section 4.4.2), when the gravity direction is known, only a 2D rotation – or one degree of freedom – and a 3D translation need to be estimated. It is also possible to remove the last degree of freedom pertaining to the rotation by exhaustive search on a 1-dimensional parameter space, spanning the rotation angle on a 2D plane. Error cones can thus be produced, and, by adding a constraint on the camera height, a conic section – more precisely, the projection of a continuous range of conic sections onto the ground – defines if the 3D correspondence is an inlier for the model. As a further step, the camera pose voting approach proposes to substitute the optimization step with a voting approach. Therefore, every match casts a vote for a limited set of camera poses it is compatible with, and the pose with the most votes yields a group of inliers for the final estimation.
To better understand the voting procedure, it is useful to reformulate the outlier rejection step of City-Scale Localization (cf. Section 4.4.2) with a further switch of perspective, moving uncertainty from the 3D points location to the camera pose itself. Indeed, any match will constrain the pose just as the pose restricts the location of a 3D point to an error cone in space. Interestingly, when passing from points to pose, the camera location

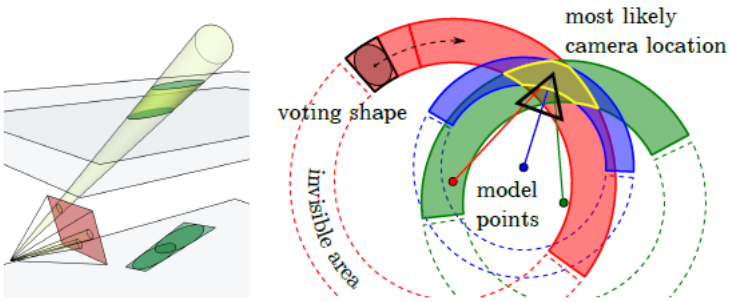


Figure 4.7: An illustration of voting shapes. On the left: the camera pose hypothesis generates error cones whose projections are then used to define error regions where we expect to find matches. On the right: the same error region is transferred to the camera centre and orientation, and votes are accumulated from all matches, defining an optimal configuration. Figure from [49].

can be shown to lie in a region with the same shape as the projected error region of a 3D point from Section 4.4.2. The necessary assumptions here are that both references are gravity aligned and the camera orientation has been temporarily fixed. The new regions are called *voting shapes*. Building on this observation, it is easy to extend the reasoning to one more degree of freedom induced by the rotation: the voting shapes are merged to form an annulus, which can also be simplified if visibility assumptions are added. The situation is illustrated in Figure 4.7. In particular, voting shapes can be observed on the right side as the juxtaposition of voting shapes pertaining to fixed camera poses

The computation of the voting shapes is carried out for all matches, and for simplicity the space is quantized into fixed size bins. Every voting shape covering a bin casts a vote for it, so that eventually the regions with the most votes can be selected. For example, in Figure 4.7 all three depicted matches cast a vote for the yellow region, which is eventually chosen as the most likely camera location. This, in turn, defines inliers which are eventually used for the final pose computation.

An interesting observation is that this voting mechanism can handle a multi-modal distribution, and is thus helpful in case of repeated structures. However, this approach implicitly assumes a large number of inliers can be found, so that the votes cast for the model stand out from the background noise. In case of match scarcity, this assumption is harder to verify, and thus the method performs worse.

Overall, camera pose voting and City-Scale Localization (cf. Section 4.4.2) set the theoretical framework for the semantic consistency score of [44]. We will deal with it in detail in Section 4.6.2. We first need to introduce a further field of research, namely the range

of methods specific to the long-term localization problem.

4.5. Visual localization with semantics

Several localization methods have tried to take advantage of additional information that is orthogonal to appearance to improve the performances of localization. Semantics are particularly interesting in this sense, because they are relatively easy to obtain and robust. In the following, we explore relevant solutions proposed in the visual localization literature with semantics.

4.5.1. Pose estimate with semantics only

To demonstrate the richness of the semantic information when performing camera pose estimation, [43] explore the localization performance that can be obtained when only using semantic information to solve for the camera matrix P.
Their approach consists in constructing a loss function in P, that is a measure of the difference of the semantic content of the query image and a reference 3D point cloud reconstructed via a standard SfM pipeline and enriched with semantic labels. Semantics are in both cases obtained by using a segmentation CNN previously trained on the Cityscapes dataset [9]. No descriptors are necessary in this method, besides when performing SfM.
The loss function is constructed by using the projection through the unknown value of P of two types of primitives – 3D points and 3D curves separating two different semantic classes. Ideally, the former should provide high level information, allowing to exclude similarity among places with very different semantic content. The latter provide more discriminatory information, as semantic labels are not considered just on standalone points, but in relation to other points with the same label and neighboring classes.
Consequently, the loss function is composed of two parts: a weighted average of the distance $d_{L_i}(P\mathbf{X}_i)$ of the projection $P\mathbf{X}_i$ on the query of a point $\mathbf{X}_i$ to the closest point sharing the semantic label L_i, and the integral of the distance $d_{L_i^1 L_i^2}(\mathbf{x}(s))$ along the projection of a curve onto the query, $P\mathcal{C}_i$, with the closest curve separating the same labels L_i^1 and L_i^2. One further refinement consists in truncating both distances at a maximum value, to avoid outliers when an object is missing in the query due, for instance, to occlusion.
Finally, the loss is computed as the following function of P:

$$E(P) = \sum_{i=1}^{N} \lambda_{L_i^1 L_i^2} \frac{1}{l_i} \int_{P\mathcal{C}_i} \eta_{L_i^1 L_i^2}(\mathbf{x}(s) ds + \sum_{i=1}^{M} \gamma_{L_i} \frac{1}{M_{L_i}} dL_i(P\mathbf{X}_i).$$

where $\lambda_{L_i^1 L_i^2}, \gamma_{L_i}$ are weights that allow different treatment of different classes, l_i is the length of the projected curve $\mathcal{C}_i$, and M_{L_I} the amount of points with label L_i in the query. The distances $dL_i, \eta_{L_i^1 L_i^2}$ are the truncated Euclidean distances.
Rather than minimizing this highly non-convex function directly, the chosen approach is to perform marginal optimization for the rotation parameters first, followed by the camera centre. Overall, the optimization strategy is quite scene-specific, and the choice of curves to project onto the query is also very much connected to the content of the scene. Hence, the procedure is not very well suited to automatization. For this reason, as well as a high variability in the accuracy of the resulting pose, the approach was later only used as theoretical justification for including semantic information in the localization pipeline.

Several methods, conversely, make use of semantics as an additional step, to enhance localization capabilities. These may be divided according to the stage in the localization pipeline where semantics are used: either in the description phase, with the goal to find a more robust yet discriminative embedding for descriptors, or in the match outlier filtering phase. In the latter case, semantics are used to enable successful termination of RANSAC even in presence of significant amounts of match outliers, or as a strategy to re-rank top retrieved images when performing coarse-to-fine localization.

4.5.2. Keypoint description with semantics

One first example of usage of semantics in the description phase may be found in the work of [1], whose method became known as semanticSIFT. The authors observe that while descriptors per-se are prone to ambiguity of appearance, matching quality may be improved by also considering the semantic content of the described patch. For example, a patch including elements of {sea, sky} would not be matched to a patch that is visually similar but contains pixels from the classes {sky, road, vegetation}, as it would be immediately clear that the two patches do not represent the same place.
To exploit the additional semantic cues, the authors adopt a bag-of-visual-word approach (cf. Section 4.2), and augment the SIFT-based vocabulary to as many copies of it, as there are possible combinations of semantic content in the patch (for instance {{sky}, {road}, {vegetation}, {sky, road}, {sky, vegetation}, {road, vegetation}, {sky, road, vegetation}, {unknown}, {sky, unknown}, ...}).
An incoming descriptor will only be matched to words which have a similar semantic content, decreasing the amount of false matches that are formed in the first place.
Although simple, this idea is very powerful, because it automatically enforces semantic consistency *before* matches are created. We will adopt a similar approach in our method, since we wish to increase the quality of matches to fight match scarcity.

One further interesting possibility when adopting a mixed visual and semantic vocabulary is to ignore the visual words that have uninteresting semantic content, as for example the class {animal}. As we will see, it will not be possible to do any such thing in our case, since segmentation will not be semantic. However, it would be interesting to find whether some classes are correlated to transitory objects, in which case they should rather be discarded.

4.6. Robust long-term visual localization

As anticipated in the problem formulation (cf. Section 3), one of the most challenging tasks of long-term visual localization is matching descriptors across very different visual conditions, as appearance is not robust in this case. The negative impact of contaminated matches on pose estimation is both in terms of computational effort and probability to output a correct pose. Indeed, match scarcity entails the correct model hardly achieves higher consensus than randomly sampled models. Hence, two strategies have been studied to improve the quality of matches: restricting the search space for descriptors, or relaxing matching criteria and subsequently pruning outliers in an additional filtering step. While in this work we will mostly be concerned with the latter option, it is worth to consider some examples from the former, too. We now illustrate some notable literature from both categories.

4.6.1. Coarse-to-fine localization

Restricting the search space of descriptors during matching can be useful for computational savings, as well as to reduce the amount of correct matches not found due to ambiguity. Indeed, it is often the case that large-scale reconstructions include similar keypoints for far-away places, which are usually not assigned to a match since there is high probability to accept gross mismatches. When restricting the search to a smaller portion of the point cloud, the probability of collisions is much lower, hence more correct matches can be retrieved.
Many of the aforementioned localization tools count on visibility information to select the most promising places for the search. Among these, P2F [23] matches from 3D to 2D and uses co-visibility with correct matches to prioritize the search (cf. Section 4.3.2), Active Search [34] filters out points that are further than two edges from the trigger point in the visibility graph when matching from 3D to 2D (cf. Section 4.3.2), and Hyperpoints [35] casts the search of the maximum number of co-visible matches as an image retrieval

problem.
Finally, HF-Net [31] combines the power of NetVLAD [3] and SuperPoint [11] in a coarse-to-fine retrieval approach to obtain state-of-the-art performances in visual localization. The pipeline first retrieves with NetVLAD database images similar to the query, and then matches to the selected places only through SuperPoint descriptors. Although this method achieves outstanding speed and accuracy, it heavily relies on learning and is therefore data-intensive. In our work we will rather focus on improving the robustness of non-learned localization methods to long-term visual variations.

4.6.2. Outlier filtering strategies

To address the scenario of relaxed matching criteria, namely a Lowe's ratio test with a rather loose tolerance (for example fixing the threshold to 0.9), both methods for deterministic and stochastic outlier filtering have been extensively studied.
The former take a hard decision on removing outliers before robust pose estimation, while the latter use a measure of the quality of matches to prioritize some over the others during pose estimation.
Among deterministic filtering, we may find simple semantic consistency [22]. Matches whose points do not have the same semantic label, or are labelled with transitory classes, are discarded.
Conversely, when using a probabilistic approach one may penalize sampling of poor-quality matches within model estimates, which falls into deterministic filtering if the sampling probabilities are set to zero. In the following, we explore some interesting methods related to this approach.

Geometric-semantic match consistency

The probabilistic outlier filter by [44] proposes to incorporate global information from the semantics of the point cloud into the sampling probabilities of each match, so to promote the sampling of semantically consistent matches and speed up the convergence of RANSAC to the right solution. The paper borrows the set up of [49] (cf. Section 4.4.3), with known gravity direction and camera height. Additionally, rather than considering error shapes for each match, they employ an exact geometry, since the constrained pose is used for computing the score and not for direct inference on the pose.

To assess the global semantic consistency, 3D points need to be projected onto the image plane through some pose. Thanks to simplifying assumptions of known gravity direction

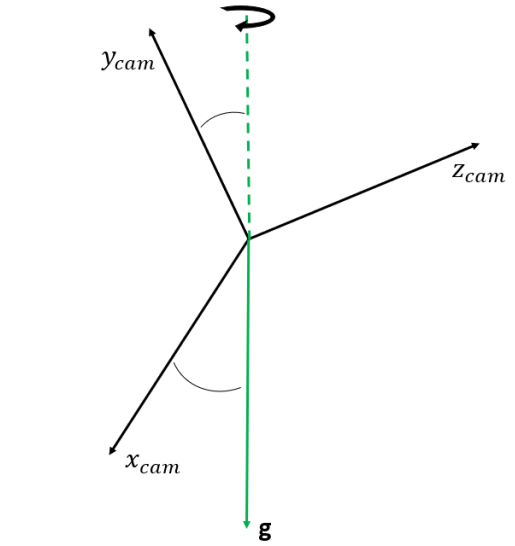
and camera height, the authors show how to lock all but one degree of freedom in the pose with the information of an individual match. The remaining uncertainty is the location of the camera centre on a circle of height z_0, with known orientation. This set of poses can be easily explored through discretization.
The evaluation is made possible by the constraint of poses to a bounded, one-dimensional parametrization thanks to the addition of simplifying assumptions of known gravity direction and camera height. We now briefly describe the reasoning to obtain these poses.

Any rotation in 3D spaces can be represented by three elemental rotations, i.e. rotations around some axis. In the GSMC derivation, the known gravity direction locks two of the three degrees of freedom of the camera rotation matrix R, since it defines the elemental rotation around two coordinate axes, only leaving ambiguity on the horizontal rotation (cf. Figure 4.8a). Moreover, the match itself, together with the gravity direction constrains the camera centre to lie on a cone with opening angle θ as measured between the ray through the considered image point and camera centre, and gravity (cf. Figure 4.8b). When the camera height is also known, the cone is reduced to a circle, in red in the figure. Finally, the last degree of freedom of the camera rotation is locked by imposing that the 3D point projects onto the 2D image point. In summary, the poses to be spanned are those around a specific circle, which can be easily parametrized through an angle $\phi \in [0, 360°)$ and explored in discrete steps.

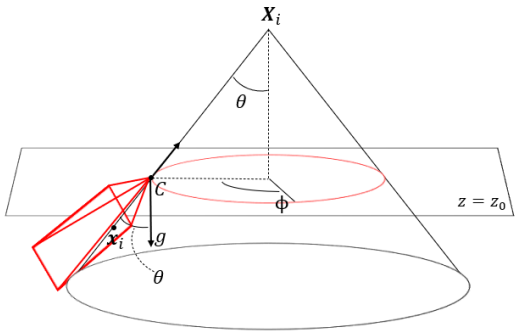
Once a finite number of poses has been selected for exploration, the set of 3D visible points from the putative camera centres is projected onto the image plane through each pose. Note not all points are projected, as this might introduce several false detections. Instead, points are filtered both through their distance from the 3D matched point and the consistency with database information of the direction and distance of observation from the putative camera centre. The final step consists in verifying which points project onto pixels that share their semantic label. These are defined *semantic inliers*, whose set we will call $\mathcal{I}_i^s \subset \mathcal{X}$. The score associated to a match is computed as the largest semantic inlier count among spanned poses, that is

$$s_i = \max_{\phi} |\mathcal{I}_i^s(\phi)|.$$

The paper shows improved localization performances across many scenarios and datasets. The semantic score is quite rich in information, as it carries an estimate of the pose for every match, and yet does not appear to be exploited to the fullest, being employed as weak signal during pose estimation. One possible reason for this is the little discriminative ability of the segmentations, which only use few semantic classes as available



(a)



(b)

Figure 4.8: Illustration of the geometry constraints induced by a match and by the known camera height and gravity. Figure (a) shows the constraints imposed on rotation of the camera axes by the known gravity direction. In the image, the angles between x_{cam} and the vertical direction, as well as y_{cam} and the vertical direction are measured and hence fixed. The remaining degree of freedom is a rotation around the vertical axis. Figure (b) instead depicts the set of possible camera centres C, laying on a cone whose vertex is $\boldsymbol{X}_i$. The angle θ as measured between $\boldsymbol{g}$ and the ray of the image point $\boldsymbol{x}_i$ determines the cone opening angle. The known camera height further reduces the possible poses to a circle (in red). This can be parametrized through the angle ϕ. For every camera centre hypothesis, the camera orientation is also fully determined. This is because of the considerations of (a) and since the position of $\boldsymbol{x}_i$ in the image plane has to align to the line between C and $\boldsymbol{X}_i$. In red we show an example of possible camera orientation.

in off-the-shelf semantic segmentation networks, and these are not explicitly trained for robustness to long-term appearance changes. Such qualities can be achieved with an appropriate training procedure, which uses 2D-2D correspondences across different seasons to enforce consistent behavior in various scenarios. We now explore how to obtain such correspondences, and later the training procedure for obtaining cross-seasonal robustness in semantic consistency.

Robust semantic segmentation across seasons

To enforce robustness of semantic segmentation across various visual conditions, it is necessary to have ground truth information on the corresponding keypoints in different scenarios. Such information can be obtained with minimum supervision, according to [27].

The paper uses the CMU-Seasons [4] and Oxford RobotCar [25] datasets with their different traversals (seasonal for CMU, day-night and seasonal for RobotCar) to generate dense point clouds for every traversal. Then, the point clouds are aligned to common world coordinates, and they proceed to search point correspondences among two point clouds by selecting mutual nearest neighbor pairs of points. Correspondences are stored for image pairs.
The approach is largely independent of the appearance of points, and can therefore achieve better invariance to long-term changes. A direct application explored in the paper is the training of a segmentation network whose labels are robust to visual variations. Let us set the notation for the training procedure, which uses a PSPNet architecture [51] to predict semantic classes for the input image. We will always assume that the correspondences are available between a reference condition – images taken with favorable weather and lighting – and a target traversal, which may include seasonal and illumination changes. We indicate the given correspondences with $(I^r, I^t, \mathbf{x}^r, \mathbf{x}^t)$, where I are the whole images and $\mathbf{x}$ are the pixel locations of corresponding points. Moreover, a standard semantic segmentation dataset with ground truth labelling is available. For the present paper, the Cityscapes dataset was chosen [9].

The segmentation network trains shared weights on a mixed dataset with Cityscapes images, which can be trained with full supervision but no correspondences, and CMU or RobotCar images, which are only weakly supervised by the correspondences. At every epoch, the network weights are updated through a mixed loss with the following terms:

$$\mathcal{L} = \mathcal{L}_{sup} + \lambda \mathcal{L}_{corr}.$$

$\mathcal{L}_{sup}$ is a standard categorical cross-entropy, while the correspondence loss $\mathcal{L}_{corr}$ is more involved. The paper proposes two versions to encourage the same prediction for corresponding points. The first is a cross-entropy for the traversal predictions on the reference labels, which quite intuitively penalizes all predictions assigned to different classes:

$$\mathcal{L}_{corr} = -\sum_{(r,t)} \frac{1}{N_{corr,(r,t)}} \sum_{i=1}^{N} \mathbf{c}_i^r \cdot log\, \mathbf{d}_i^t,$$

with $N_{corr,(r,t)}$ the number of correspondences for a given reference-traversal image pair (I^r, I^t), $\mathbf{c}_i^r$ the one-hot assignment of correspondence i and $\mathbf{d}_i^t$ the feature in the last CNN layer describing the point $\mathbf{x}_i^t$.
The second proposed loss is a hinge loss, whose form is

$$\mathcal{L}_{corr} = \sum_{(r,t)} \frac{1}{N_{corr,(r,t)}} \sum_{i=1}^{N} max(0, m - \frac{\mathbf{d}_i^r \cdot \mathbf{d}_i^t}{\|\mathbf{d}_i^r\| \|\mathbf{d}_i^t\|}).$$

$N_{corr,(r,t)}$ is the number of correspondences for a given reference-traversal image pair (I^r, I^t), while $\mathbf{d}_i$ are the last- or second-last-layer feature vectors in the segmentation CNN describing the pixels from the i-th correspondence. This loss pushes the feature vectors of reference and traversal points to become parallel – and consequently yield the same prediction at inference time. The authors propose a margin of 0.8, which allows for an angle of about 37° between the two vectors.
Training is further refined by excluding the correspondences belonging to transitory classes, like cars or pedestrians. The rationale for this correction is that the correspondence must either be mistakenly assigned or uninteresting for localization purposes, since the objects are free to move in the scene.

Overall, the value of the correspondence dataset is greater than just a means for performing domain adaptation. Indeed, it is possible to design a self-supervised training procedure to produce much more fine-grained segmentations, and enforce consistency across visual variations. We now discuss the theoretic background for this training procedure, named *Deep Clustering*.

Deep Clustering

Convolutional neural networks have proven excellent for learning image-based tasks, but their weakness lays in the domain specificity. In an effort to train general purpose features, the authors of [6] study techniques to scale model training to billions o images, which are easily found online. In this challenging scenario, it is not possible to annotate every image

manually. Therefore, a self-supervised training procedure must be studied.

The starting point is the observation that convolutional neural networks retain strong prior on the input, and for this reason they tend to perform well even before training on classification tasks where inputs are well separated. In the case of a classfication task on the ImageNet dataset [10], the authors report an accuracy of 12%, far above the 0.1% of random labelling. They thus propose to bootstrap the discriminative ability of the network with a succession of clustering of the final layer features and classification to the pseudo-labels obtained through clustering. Figure 4.9 illustrates the training procedure adopted in the paper.

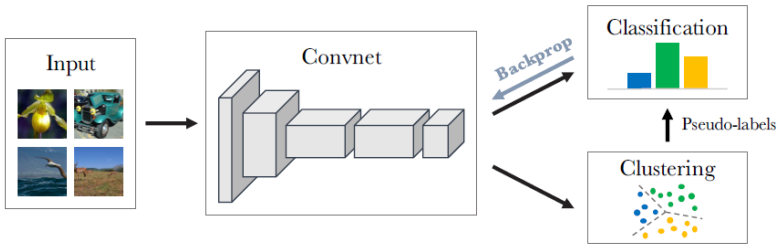


Figure 4.9: Illustration of the pipeline of Deep Clustering. The same network is used to produce features to be clustered and for predicting the pseudo-labels obtained at the previous clustering step. Figure from [6].

Formally, assume the encoding network is a mapping $\mathbf{d}_\theta(\cdot) \in \mathbb{R}^D$ depending on weights θ. The purpose of the paper is to find a mapping $\mathbf{d}_{\theta^*}(\cdot)$ whose embedding is well suited for a wide range of tasks, like classification, image retrieval, ...
Assuming the specific task is represented by a further mapping g_W on top of the previous features $\mathbf{d}_\theta(\cdot)$, given N input images $\{I_n\}$ and target labels $\{y_n\}$, a cross-entropy loss is employed to train the features $\mathbf{d}_\theta(\cdot)$:

$$\min_{\theta,W} \frac{1}{N} \sum_{n=1}^{N} l_{CE}(g_W(\mathbf{d}_\theta(I_n)), y_n)$$

with $l_{CE}(\mathbf{p}, y) = -\sum_{k=1}^{K} log\, p_k\, \mathbf{I}_{y=k}$ and $\mathbf{I}_{(\cdot)}$ is the indicator function.
The proposed method alternates on every epoch training weights (θ, W) to the above loss and finding suitable labels $\{y_n\}$ through k-means clustering of $\{\mathbf{d}_\theta(I_n)\}$.
To assess what the network has learned after training, the authors report the 9 database images that cause the highest excitation of a few selected filters in the final convolution of the encoding network (a standard AlexNet architecture [20] is used for simplicity). As depicted in Figure 4.10, the network seems to have learned to map different object classes

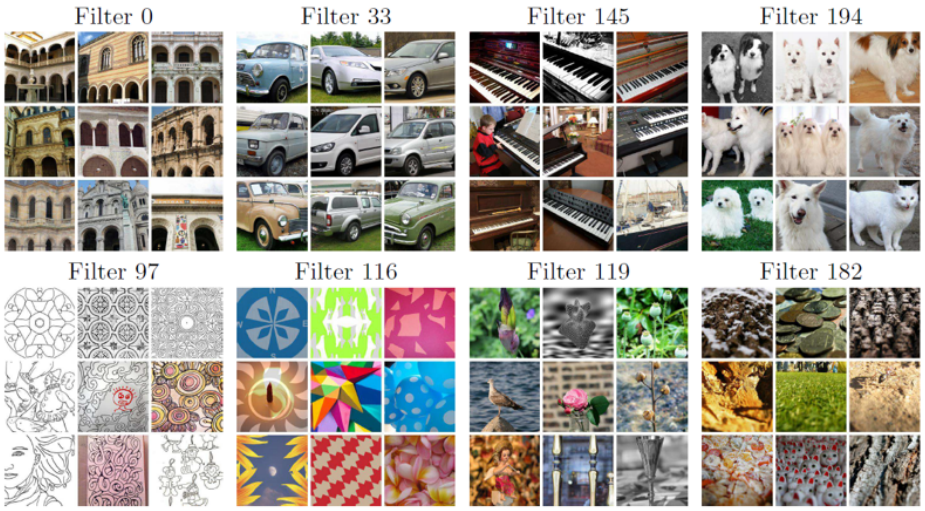


Figure 4.10: The top 9 activated images for six filters in the last convolutional layer of the network, which directly produces the features $\mathbf{d}_\theta$ to be used for clustering. The top row shows filters that are maximally excited by objects of the same kind, while the bottom row filters respond to the same styles. Figure from [6].

to different filters, which are in a one-to-one correspondence with the D components of $\mathbf{d}_\theta$. In particular, the top row filters gather images from homogeneous object classes, while at the bottom stylistic textures are more impactful. Since the features are then commonly used as scores for classification, the visualization results suggest the network learned to classify similar objects in the same classes as a subproduct of the self-supervised training procedure.

The final assessment on ImageNet [10] places classifiers trained with Deep Clustering only 1.4% in accuracy below the benchmark of supervised networks. This result confirms the validity of the method. Furthermore, the general nature of the learned features is verified by evaluating the network on different data (PASCAL VOC datasets). The evaluation sets Deep Cluster well above the other unsupervised methods and about 6.9% below supervised networks trained on ImageNet.
The results of this paper are very encouraging towards the possibility of obtaining self-supervised classes even in the problem of semantic segmentation, which is just an instance of classification, with labels assigned to every pixel rather than to the whole image. Thus, Deep Clustering opens up to the possibility of segmenting far more classes than it is usually possible with human annotators. In our problem, for example, it might allow to distinguish not only vehicles from vegetation, but also wheels from car body and trees from bushes.

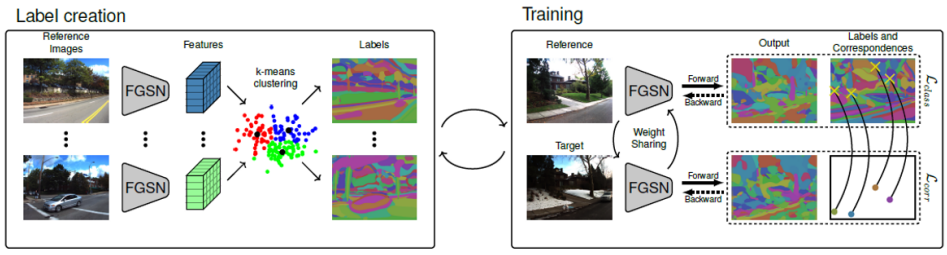


Figure 4.11: Fine-Grained Segmentation Networks is trained in a self-supervised manner to classify pixels in a pre-defined number of classes. Several repetitions of clustering of features, followed by training the network to predict the cluster labels are performed for training. At training time, robustness across visual variations is enforced by means of geometric correspondences mapping a reference condition to a variety of other seasonal conditions. Figure from [22].

Fine-grained segmentation networks

Starting from the previous correspondence dataset (cf. Section 4.6.2), and leveraging the results of Deep Clustering (cf. Section 4.6.2), [22] propose an extension of the semantic consistency score of [44] (cf. Section 4.6.2). The rationale is that few semantic classes, as allowed from supervised segmentation networks, have a small discriminative power, especially in urban scenarios where only few semantic classes appear with several different instances. The semantic consistency score needs a large variety of classes in the scene to increase its ability to select the correct matches. Conversely, it is not required that classes hold semantic meaning, as long as the class assignments are repeatable. Indeed, the desired property of high quality matches is stability under visual changes, just as shown by the success of SuperPoint (cf. Section 4.2).
In light of these observations, the authors propose to train a custom number of robust semantic classes by enforcing the same class prediction on corresponding points across visual conditions. Such network is named *Fine-Grained Segmentation Network* (FGSN). Let us now describe its architecture and training procedure.

The network is largely inspired on the architecture employed for the robust semantic segmentation training with correspondences (cf. Section 4.6.2). The base is a PSPNet [51], which produces features that are later fed to a dense layer for pixel-wise classification. The network is pre-trained on fully supervised semantic segmentation, using the Cityscapes [9] and Mapillary Vistas [28] datasets, among the best-known datasets for semantic segmentation. The subsequent training procedure, depicted in Figure 4.11 alternates clustering the pooled features to obtain pseudo-labels and training the network weights under the supervision of those labels, as in Deep Clustering (cf. Section 4.6.2). The clustering

step is repeated every 10000 epochs, followed by random re-initialization of weights in the classification layer. The dataset used at this stage are the CMU Seasons Correspondence Dataset and Oxford RobotCar Correspondence Dataset from [27], whose images are not overlapping with the Extended CMU Seasons and Oxford RobotCar images used for evaluation. As in that scenario, correspondences are available between a reference traversal and all other target traversals.
Similar to the cross-season correspondences work, the training loss of FGSN also has two components, to encourage the correct labelling of the class in the reference traversal and to adapt the output of the target traversal to the corresponding point in the reference. The two terms are both cross-entropy losses. Let a sample pair of reference-target images be $(I^r, I^t, \mathbf{x}^r, \mathbf{x}^t)$, where I are the whole images and $\mathbf{x}$ are the pixel locations of corresponding points. The classification loss is straightforwardly computed as the average misclassification error over all pixels in the reference image. Batches with only one pair (r, t) are used for computational limits. Formally,

$$\mathcal{L}_{class} = -\frac{1}{N_{pixels}} \sum_{i=1}^{N_{pixels}} \mathbf{c}_i^r \cdot log\, \mathbf{y}_{i,pred}^r,$$

where $\mathbf{y}_{pred,i}^r$ is the vector of predicted class scores at pixel i, and $\mathbf{c}_i^r$ the pseudo-label for that pixel. The correspondence loss term is instead

$$\mathcal{L}_{corr} = -\frac{1}{N_{corr}} \sum_{i=1}^{N_{corr}} \mathbf{c}_i^r \cdot (log\, \mathbf{d}_i^r + log\, \mathbf{d}_i^t),$$

where $\mathbf{d}_i$ are the latent representation vectors at pixel i. The final loss is computed as $\mathcal{L} = \mathcal{L}_{class} + \mathcal{L}_{corr}$.

Provided the correspondences are of high quality, the FGSN can produce a fine texture of segmentation classes that are invariant to long-term appearance variations and that best reflect the dataset content. Moreover, the network needs very little human supervision for training, which makes it suitable for large scale settings.
The authors exploit the novel class predictions to compute a match consistency score, which is used as a bias towards the most promising matches at RANSAC's sampling stage, in the same fashion as [44]. This strategy achieves excellent results, increasing of several points the accuracy achieved in [44] for most settings and precision levels. However, the performance is still lagging behind state-of-the-art methods employing learned descriptors. In the Extended CMU Seasons dataset [4, 37], this is especially true for the most challenging scenario, i.e. the Park setting. Whereas urban settings offer better stability in presence of buildings, this subset of data is characterized by a predominance of

vegetation. The visual variations of long-term scenarios entail increased ambiguity in the space of descriptors and more points with no detected correspondent. While providing an excellent tool for filtering the resulting matches, the GSMC method appears to suffer from the declined match quality. Therefore, in our work we target the issue upstream, and look at improving correspondences.

5 | Proposed method

As illustrated in the problem formulation of Chapter 3, we will be concerned with localization in situations where very few matches are available, or in general largely ambiguous situations. In these settings, we choose to use semantic cues of images to overcome the challenge of estimating a correct pose. This work introduces two novel tools: a matching strategy, which we name *Semantic Matching*, and an improved version of the RANSAC [13] algorithm including *Biased Consensus* to perform effective pose estimation in situations with match scarcity.

We illustrate each of them, after discussing the innovation with respect to similar approaches.

5.1. Long-term visual variations and repeated structures

Long-term scenarios are affected by match scarcity due to two concurring effects, that are (*i*) the disappearance of some keypoints due to changed structure of the scene, and (*ii*) the variation of appearance of other keypoints, perturbing the associated location in the descriptor space. Whereas previous methods such as Geometric-Semantic Match Consistency [44] and Fine-Grained Segmentation Networks [22] create match filtering strategies based on semantics, we propose to anticipate the focus of semantic consistency during the matching process, to benefit maximally from the richness of information that semantics carry. In fact, if those methods look at increasing the chances of sampling the correct matches during pose estimation, we work in situations where filtered matches may not be enough to achieve consensus above the level of chance. Hence, our Semantic Matching approach is designed to recover a crucial portion of matches, in addition to increasing their quality, rather than filtering potentially valid matches.
Because a portion of matches lost to long-term variations are ambiguous correspondences, we learn lessons from the problem of repeated structures. Among those works, the Hyperpoint strategy [35] embraces the ambiguity by postponing the formation of definitive

matches, and work with candidate matches to find the combinations that allow the best solution globally, while the camera pose voting method [49] highlights the importance of the broader context to solve the local ambiguity.
We assume that several correct matches lay within the first few neighbors, but not necessarily the closest. Similarly to the camera pose voting approach, we use context information orthogonal to appearance to guide the decision on which matches we should retain. However, in contrast to [35], we do not employ a retrieval approach, as match scarcity makes it hard to retrieve the correct images in the first place. Moreover, whereas [49] employ a voting procedure without choosing any particular match above other candidates, our setting with match scarcity dictates a more surgical approach. We thus use global semantic consistency of candidate matches to select a single best match.
We also need to cope with the increase of interest points with no correspondent, whose inclusion in the set of matches is adding complexity to the pose estimation stage, and could potentially disrupt the localization overall.
It is difficult to set these points apart from matchable keypoints, also because of the blurred boundary from keypoints with a noisy location. In fact, as noted by [11], the notion itself of a keypoint is ill-defined. For these reasons, most methods in the visual localization literature avoid discussing the nature of keypoints and postpone actions to already formed matches, where the distinctive quality of descriptors and robust fitting (e.g. RANSAC [13]) take care of the filtering. Similarly to [44], our method addresses the issue in a soft manner, cutting down the probability of sampling keypoints with no correct match through semantic consistency. Thanks to an improved score computation, our method is more effective than the original Semantic Consistency score when reducing the impact of false matches.

5.2. Semantic Matching

Figure 5.1 illustrates the functioning of Semantic Matching. Following [33] we match from query to database keypoints. The query keypoints are represented in the figure by yellow and red stars, and we may see the database descriptors associated as nearest neighbors (pink and purple dots) do not correspond to the correct match. However, the correct descriptors might be within the first few neighbors and we look to retrieve them through semantic consistency.
Let us define *features* the query descriptors and associated keypoints and *points* the database descriptors and 3D location. For every feature $\boldsymbol{f}_i$, we select a neighborhood $\mathcal{U}_i$ in the descriptor space, and search for a matching descriptor $\boldsymbol{d}_{k^*}$ among those falling into this neighborhood. The neighborhood might include a fixed or variable number of

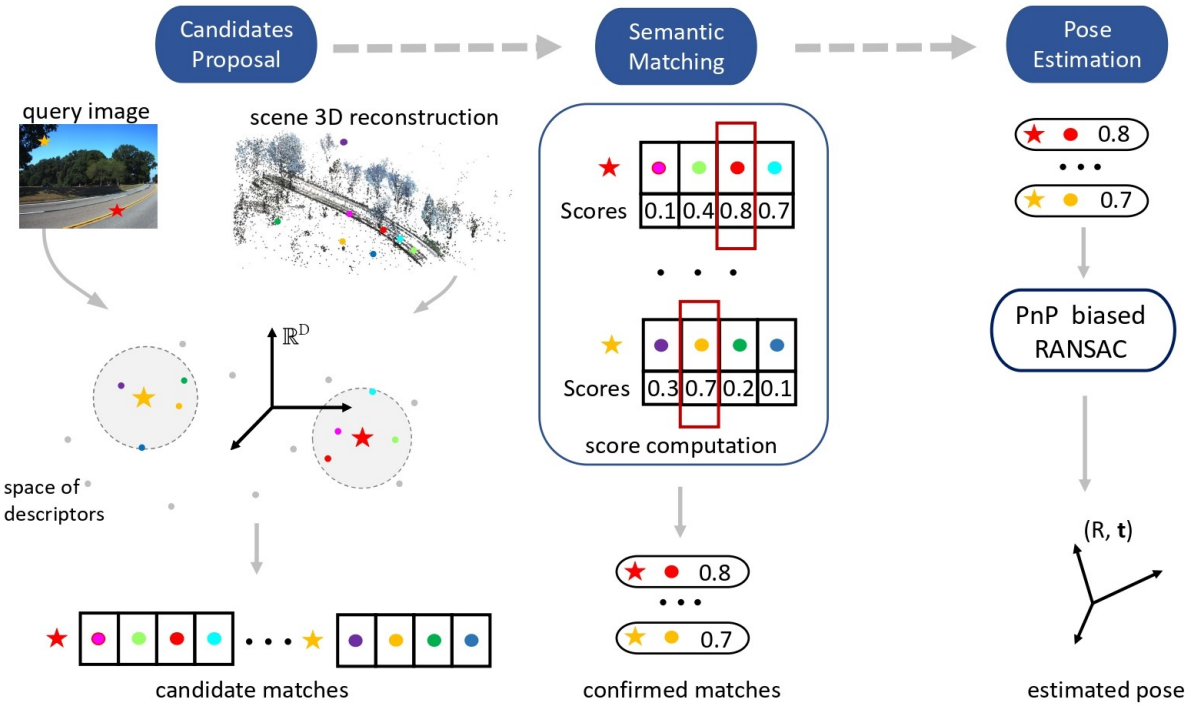


Figure 5.1: Illustration of our method. In a first phase, keypoints are associated to a neighborhood of 3D points based on appearance. The best candidate is then selected based on the semantic score. Finally, all information is used to fit a final pose estimate.

neighbors. In the latter case, for example, it may depend on the local density of the descriptor space, such as words in a visual vocabulary framework. In our experiments we select a fixed number of neighbors K. The so obtained pairs form *candidate matches*, among which we select exactly one confirmed match. To this end, we compute a quality measure q associated to each candidate match, and select the match with highest quality. Formally,

$$\boldsymbol{d}_{k^*} = arg \max_{\boldsymbol{d}_k \in \mathcal{U}_i} q(\boldsymbol{d}_k).$$

As we highlighted above, local appearance alone is often ambiguous in presence of long-term variations. The quality measure q should be designed to bring in orthogonal information, so to single out correct matches. To this end, we propose to use a fine-grained semantic score inspired by the works of [22, 44]. The rationale is that projecting the whole scene through a single match injects global information into the choice of the correct match. A fine grained segmentation helps making scores more distinctive.
Similarly to [44], we take as inputs the gravity direction in the camera reference system and with known camera height z_0, and use the same derivation of the camera pose hypothesis by spanning the centres positions along a circle with angle ϕ and finding the

camera reference system accordingly. We finally project semantic labels onto the query semantic mask, both obtained with a fine-grained segmentation network as in [22], to obtain semantic inliers $\mathcal{I}_i^s \subset \mathcal{X}$, and overall projected points $\mathcal{P}_i^s \subset \mathcal{X}$. Unlike [22, 44], we choose to weigh both the correct and incorrect semantic projections in the semantic score. Indeed, we observe that commonly appearing classes, such as greenery in scenes with predominance of vegetation, may achieve high numbers of semantic inliers even in mistaken poses. In those cases, less frequent classes would still achieve lower scores, which supports the idea of including this information into the score (see for an example Figure 6.13). The proposed score will thus be the ratio of semantic inliers to the sum of semantic inliers and outliers, that are projected points:

$$q = \max_{\phi} \frac{|\mathcal{I}_i^s(\phi)|}{|\mathcal{P}_i^s(\phi)|}.$$

A summary of our matching procedure can be found in Algorithm 5.1. We input a set of query keypoints and descriptors $\mathcal{Q} = \{(\boldsymbol{x}_i, \boldsymbol{f}_i)\}_{i=1}^{N_q}$, the query semantic mask M, the point cloud points and descriptors reconstructed from database with additional semantic and visibility information (visibility direction and angle, distance boundaries) $\mathcal{X} = \{(\boldsymbol{X}_j, \boldsymbol{d}_j, c_j, \boldsymbol{v}_j, \theta_j, d_j^{low}, d_j^{up})\}_{j=1}^{N_X}$, the gravity direction $\boldsymbol{g}$ and the camera height z_0. The algorithm produces a set of matches $\mathcal{M} = \{(\boldsymbol{x}_i, \boldsymbol{X}_i),\ s_i,\ (R_i, \boldsymbol{t}_i)\}_{i=1}^{N_q}$, each associated with a score and the pose which yielded the score.

Algorithm 5.1 Semantically guided matching

Inputs:

$\mathcal{Q}$	query keypoints and descriptors
M	query semantics
$\mathcal{X}$	point cloud with descriptors, semantics, visibility
$\boldsymbol{g}$	gravity direction
z_0	camera height

Output:

$\mathcal{M}$	match set with scores and poses

$\mathcal{M} \leftarrow \emptyset$
for all $i = 1, ..., N_q$ **do**
  $\mathcal{U}_i \leftarrow \{\boldsymbol{X}_1, ..., \boldsymbol{X}_K\}$ {select K nearest neighbors in descriptor space}
  $\mathcal{M}_{i,K} \leftarrow \{(\boldsymbol{x}_i, \boldsymbol{X}_k)\}_{k \in \mathcal{U}_i}$ {Form candidate matches set}
  for all $(\boldsymbol{x}_i, \boldsymbol{X}_k) \in \mathcal{M}_{i,K}$ **do**
    $\mathcal{I}^s, \mathcal{P}^s, (\mathrm{R}, \boldsymbol{t}) \leftarrow \text{GSMC Score}(\boldsymbol{x}_i,\ \boldsymbol{X}_{j,k},\ \boldsymbol{g},\ z_0,\ \mathrm{M},\ \mathcal{X})$
    $s_{i,k} \leftarrow \frac{\mathcal{I}^s}{\mathcal{P}^s}$
  end for
  $i^* \leftarrow argmax_{\mathcal{U}_i} s_{i,k}$
  $\mathcal{M} \leftarrow \mathcal{M} \cup \{(\boldsymbol{x}_{i^*}, \boldsymbol{X}_{i^*}),\ s_{i^*},\ (R_{i^*}, \boldsymbol{t}_{i^*})\}$ {Add confirmed match to final result}
end for=0

Finally, we use the computed matches directly for pose estimation without filtering. We adopt the biased RANSAC framework proposed by [44]. The overall pipeline can be observed in Figure 5.2.

While increasing the computational cost of creating matches, the semantic score is rich in information that may be used also in subsequent steps of localization. The semantic score may be used as a sampling probability during pose estimation, as shown in [44]. Moreover, as we show in the experimental section (6.2.6), the score can be used as weight to evaluate model consensus, with gains of over 10% of correctly localized images in severely ambiguous scenes. Finally, we also note that the score computation outputs as a by-product an estimate for the pose of the camera. One interesting future research direction aims to explore pose inference directly on these poses.

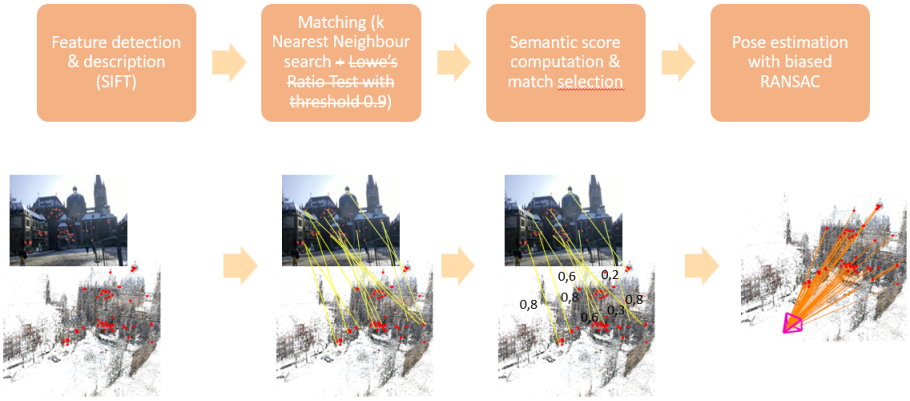


Figure 5.2: Illustration of our proposed pipeline. After describing inputs, all k nearest neighbors of query descriptors are evaluated, based on the semantic score of the resulting candidate match. Chosen matches and their scores are then used to estimate a pose with biased RANSAC. Figure adapted from [32].

5.3. Robust fitting with match scarcity

Another weakness of visual localization with long-term settings rests in the robust pose estimation algorithm that is commonly used, that is the RANSAC [13] framework.
To estimate a pose with robust fitting, a RANSAC [13] iteration comprises three phases: (i) sampling correspondences in number equal to the minimal sample size (MSS), (ii) estimation of a hypothesis of pose θ with the PnP method on the sampled points, (iii) evaluation of consensus.

We now focus on stage (iii): if we consider the number of inliers $\mathcal{I}$ associated to correct and random poses, the correct functioning of RANSAC rests on the assumption that $|\mathcal{I}_{gt}| \gg |\mathcal{I}_{(\mathrm{R,t})}| \ \forall \ (\mathrm{R,t}) \neq (\mathrm{R}_{gt}, \mathrm{t}_{gt})$. However, in long-term settings few correct matches $\{(\boldsymbol{x}_i, \boldsymbol{X}_i)\}$ could be available, due to *match scarcity*. Thus the assumption is likely to be violated, and incorrect poses could be accepted as better than the correct one.

5.4. Biased Consensus

The key solution to the consensus problem problem is to include once more orthogonal information to the current evaluation. Thus, instead of counting the number of inliers we will be looking at their total quality.

Consider a collection of matches $\mathcal{M} = \{(\boldsymbol{m}_i, q_i)\}$ with associated semantic score.
We propose to use the previously computed semantic scores for pose evaluation, to better assess models in presence of match scarcity. Particularly, we evaluate a model through

the total quality of matches that agree with this model. Formally, if $\mathcal{I}_\theta$ are the inliers of a pose $\theta = (\mathrm{R}, \mathbf{t})$, we evaluate consensus as $CS = \sum_{i \in \mathcal{I}_\theta} q_i$.

The outline of our fitting method can be found in algorithm 5.2.

Algorithm 5.2 Biased Sampling and Consensus

Inputs:
$\mathcal{M} = \{(\boldsymbol{m}_i, q_i)\},\ N_{iter}$
Output:
$final\theta$

$MSS \leftarrow 4,\ bestCS \leftarrow 0,\ best\theta \leftarrow \emptyset$
for $it = 1, ..., N_{iter}$ **do**
  $MS \leftarrow sample(MSS, \{\boldsymbol{m}_i\}, \{q_i\})$
  $\theta, \mathcal{I}_\theta \leftarrow PnP(MS)$
  $CS_\theta \leftarrow \sum_{i \in \mathcal{I}_\theta} q_i$
  if $CS_\theta > bestCS$ **then**
    $bestCS \leftarrow CS_\theta$
    $best\theta \leftarrow \theta$
  end if
end for
$final\theta \leftarrow local\ optimization(best\theta, \mathcal{I}_{best\theta})$
=0

The best model will only be updated if it can exhibit matches with better quality than previously sampled poses. Overall, this more conservative estimate is well suited to situations where the best model is not believed to have significantly more consensus than a random sample.

6 | Experimental evaluation

In the experimental section, we wish to empirically validate the theoretic model behind our matching strategy.
To this end, we carefully select a database with sequences of images that exhibit significant long-term variations with match scarcity, and empirically demonstrate analogies between long-term scenarios and repeated structures.
We use the selected sequences to verify the effectiveness of Semantic Matching and Biased Consensus, both individually and jointly. Finally, we are able to outperform the state of the art algorithms for long-term visual localization of over 14% of correctly localized images, with overall more than doubled performance.

6.1. Setting

In this section we introduce dataset and tools used for conducting experiments, as well as reporting the details of our implementation.

6.1.1. Data set

The dataset we use to perform our experiments is the Extended CMU Seasons dataset [4, 37]. This dataset is composed of over 100k pictures from two cameras mounted on a car, organised in 17 *slices*, i.e. separate chunks of the traversed route. Such route, running across the city of Pittsburgh (PA, USA), was traversed in 11 different dates between 01-09-2010 and 28-07-2011 to cover as many different visual conditions as possible. Captured conditions include views with and without foliage and snow, and the lighting varies with sunny, overcast, and low sun scenarios.

Slices are divided in Urban, Suburban and Park types, according to the main content of their images. In Figure 6.1 we compare some samples from Urban and Park slices. While in the Urban scenario we may find several buildings and city facilities, the Park images are dominated by vegetation. Even for a human eye, the latter present fewer reference points and get easily confused with one another. Moreover, the seasonal excursion is much more

Figure 6.1: Some images from the Extended CMU Dataset [4, 37]. The top row reports examples from the Urban scenario, while on the bottom we can see Park images.

impactful in these cases, whereas urban scenes are more stable. The reported results of other methods confirm this observation, with a decrease of 32.1% of localized images at the finest precision for the best performing method HF-Net [31].
Each slice in the dataset includes a point cloud, reconstructed from the images of a single traversal – that of 04-04-2011, which we will refer to as *database*. Additionally, all database images and some of the others, which we will define as *query*, are provided with ground truth poses. For our experiments, we only use queries with ground truth poses, so to be able to evaluate our results. All database and query images have SIFT keypoint and descriptor data.
A mapping of database image points to their reconstruction is also provided, so that it is possible to trace back the keypoint information associated to every point in the point cloud. Particularly, each 3D point can be represented with two or more SIFT descriptors from the database images it was reconstructed from. We choose to work on two slices, one from Park (22) and one from Urban (6), which offer plenty of sequences exhibiting long-term variations, match scarcity, repeated structures and a mixture of these.

6.1.2. Libraries and implementation details

To reproduce a complete localization pipeline, we implement from scratch matching and pose estimation routines, using tools from well-known computer vision libraries.
Firstly, for matching SIFT descriptors we rely on the FLANN library [26], which offers a framework for fast approximate nearest neighbor search. Differently from [22, 44], we use all database descriptors rather than the average on each 3D point.
We set a fixed number of neighboring descriptors to search for the confirmed match. We experimentally find that $k = 4$ works well in most scenarios.

To obtain the Fine-Grained Segmentation masks for scoring, we use the code from [22] and generate predictions from their trained network (100 clusters, trained on correspondences from the CMU Seasons dataset [4]).
We implement our version of the semantic consistency score with NumPy and some tools from the OpenCV library. In particular, to project the point cloud we use the ProjectPoints() function with a correction of distortion according to the Brown-Conrady model, as specified by the authors of the dataset [37].
Some additional checks are added to maintain feasible the computation of the scores, since we require to compute several times as many scores as the previous semantic consistency filter [44]. We select a $w \times w$ window around the query keypoint and check whether the candidate 3D point's semantic label appears within the labels present in the semantic window. If not, the match is assigned a score of zero. Moreover, during computation of the number of semantic inliers and outliers, we add one additional check, namely that the 3D point is visible from the spanned camera centre positions. We assess visibility as in [44]. If the point is not visible, the score is set to zero.
All k scores are then evaluated to find the best matching 3D point for every query keypoint. In case of equal scores (including zero), the closest to the query descriptor is selected.
After obtaining correspondences, we estimate a pose with a P3P solver inside a RANSAC loop. Because we experiment with biased sampling, we choose to implement our own version of the algorithm for pose estimation, using logic closely modeled on the solvePnPRansac function in the OpenCV library. In particular, we maintain the reprojection error of 8 pixels, and use the Efficient PnP solver (cv.SOLVEPNP_EPNP) for local optimization.

6.2. Experiments

We compare our method against others in literature on both the ability to produce correct matches and the quality of the resulting poses. We start by assessing matching, then perform a comparison of several versions of our algorithm to validate some design choices. Finally, we compare the localization performance of our methods against state-of-the-art methods, showing increased localization ability in most settings.

6.2.1. Evaluation metrics

In the first experiment regarding the ability to find correct matches, we wish our method to detect as many as possible, yet without having too many false matches. In fact, two

pitfalls arise when too many outliers are present: (i) the probability of sampling a correct model is very low, therefore a very large number of iterations is needed to ensure the correct model has been found with high confidence, (ii) the probability of larger consensus sets from wrong poses increases. The latter becomes a particularly serious concern when there are few correct matches, as other images with similar structures could obtain a larger consensus, as reported in [49]. For these reasons, our metrics of choice in this experiment are recall and precision of the produced matches.
Formally, we build $N_q \times N_X$-dimensional match tables indexed by query keypoints (N_q) in the rows and 3D points (N_X) in the columns, and such that the entry is 1 on the real or detected correspondences, and 0 otherwise. Let $\boldsymbol{M}_{gt}$ be the ground truth matches table, and $\boldsymbol{M}_e$ the estimated matches table. We find recall as

$$recall = \frac{TP}{TP + FN} = \frac{\sum_{i,j}(\boldsymbol{M}_{gt} \ \& \ \boldsymbol{M}_e)_{i,j}}{\sum_{i,j}(\boldsymbol{M}_{gt})_{i,j}}.$$

Similarly, precision is

$$precision = \frac{TP}{TP + FP} = \frac{\sum_{i,j}(\boldsymbol{M}_{gt} \ \& \ \boldsymbol{M}_e)_{i,j}}{\sum_{i,j}(\boldsymbol{M}_e)_{i,j}}.$$

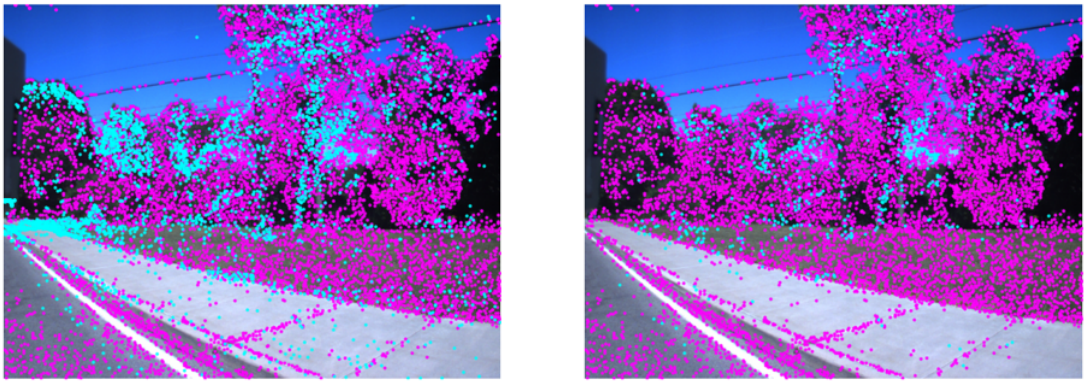
With regard to the pose estimation experiments, we use the standard evaluation framework proposed by [37]. As introduced in the problem formulation in Section 3, we compute the percentage of queries that fall within fixed thresholds of position and orientation error. We select the commonly used thresholds:

- *High precision*: 0.25m, 2°
- *Medium precision*: 0.5m, 5°
- *Coarse precision*: 5m, 10°

6.2.2. Obtaining ground truth correspondences

Some of our experiments require the availability of ground truth matches. As introduced by [11] (see Section 4.2) the task of defining interest points so to be repeatable across images is quite hard. Ideally, we interpret correspondences as the location of the same point across different images if these also correspond to the same 3D location. Note this last condition is crucial, as objects are not rigid in our scenes. To the best of our knowledge no datasets offer such ground truth information.

While having to deal with the lack of exact correspondences, a more pragmatic definition



(a) Projected points from the whole point cloud (b) Confirmed projected points

Figure 6.2: An example of projected point cloud (a) with and (b) without visibility filtering (cyan points) and the query keypoints (fuchsia points). A large density of points in Figure (a) is concentrated on the left side of the Figure (b). A subsequent filtering through visibility data shows that these points do not project onto the image. These are likely far away points whose projections fall onto the image plane, but are not visible in the image since some trees occlude their view.

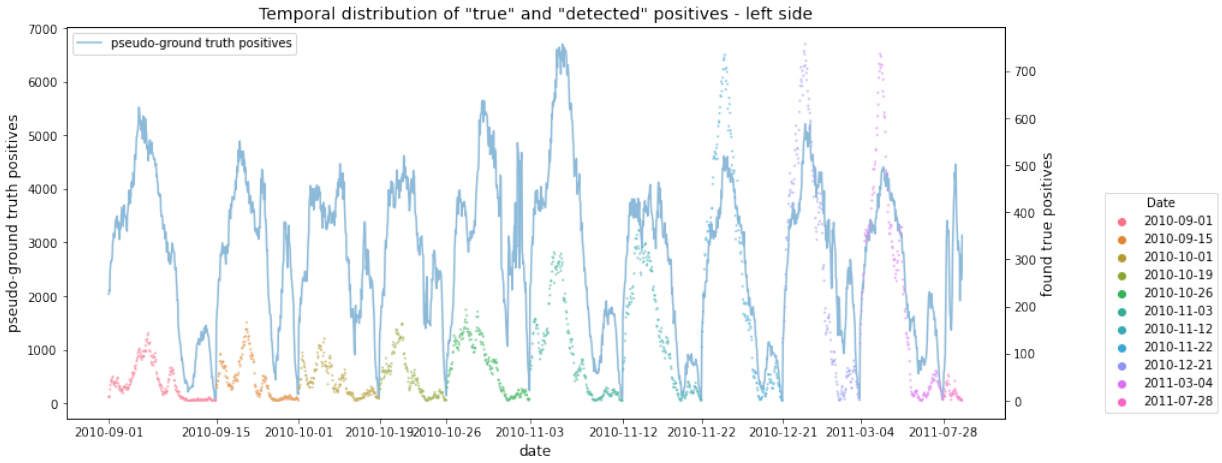
suffices to our purposes. Since we want to show we can *improve* the ability to detect valid matches, we do not make a distinction on whether the correspondences refer to the same point in the represented object, as long as their locations yield accurate pose estimates. This means that for every 3D point we will be satisfied if we can retrieve matches that fall within a small neighborhood of its projection in the image plane of the query image. To keep a conservative estimate, we choose a neighborhood of 5 pixels around the projection, which is smaller than the 8 pixels reprojection error used to evaluate model consensus. We can easily obtain all point projections with the ground truth poses from our database. Most of these pseudo-ground truth matches will never be retrieved based on appearance, as the keypoint will likely match the appearance of at most one keypoint in its neighborhood. Yet, observing the increase in the number of retrieved matches we can still effectively compare different matching strategies.

Thus, we create pseudo-ground truth matches for all queries by projecting 3D points onto the image plane. As observed in [44], not all points should be projected, as not all are visible in the picture. Consider, for example, Figure 6.2. The left side image reports all points whose projection falls into the image plane (30496). However, a large portion of these points is concentrated on the left-hand side in a small area, with several points falling on the same pixels. It is thus clear that, without further filtering, several hidden points project onto the image plane. Hence the necessity to design a mechanism

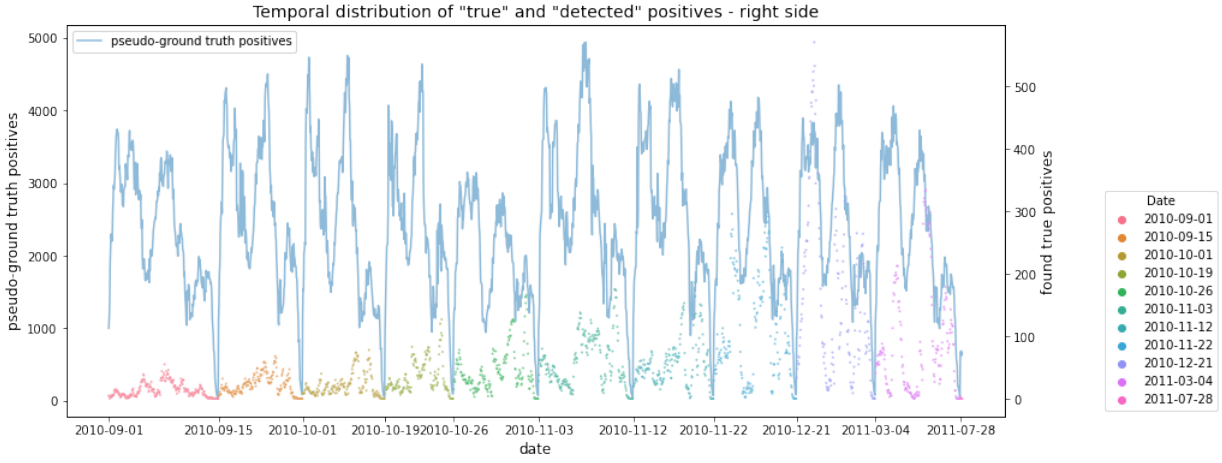
for selecting projection points. We use the visibility filtering technique adopted in [44] – we only project a point if the direction of viewing and the distance from the camera centre are within the ranges observed during reconstruction. After applying this filter, we plot the resulting points (636) in 6.2b. The concentration of points on the left has now disappeared, confirming most of the projected points of the image are not actually visible.
Despite using the visibility filter to project, we observe this might be too restrictive. Indeed, when the visibility recorded at the stage of reconstruction from the database is limited, possibly due to illumination or noise, points hardly pass the filter, despite being clearly visible. For this reason, whereas database images observe a relatively stable number of points –roughly 2000– visibility filters can lower this amount to less than 100. Thus, in the following experiments we prefer to compute ground truths on the set of seen points of the database images whose centre is within 10 m from the considered camera.

6.2.3. Choosing challenging sequences of images

Having obtained pseudo-ground truth matches, we first employ them to select sequences of images that best allow to test the performance of our algorithm.
We look for sequences that present match scarcity, long-term ambiguities and repeated structures. We initially focus our search on the Park setting, as it has been previously indicated as a very difficult localization sequence [37] due to limited capabilities of local description.
Since our strategy to find matches tends to overestimate the amount of correspondences we can detect, to pick the most appropriate sequences we rely on the joint observation of the amount of projected points, ground truth matches obtained from all such points and detected matches. For the latter, for computational reasons we stop the search at all candidate matches produced by the four nearest neighbors to query keypoints in the descriptor space. Figure 6.3 illustrates the full data for both cameras. We represent on the same chronological disposition of images the overall amount per image of pseudo-ground truth matches and detected matches, obtained considering all matches formed by query descriptors and four nearest neighbors in the database.
All trends for the left side seem more regular on the left traversal (Figure 6.3a), hence we choose to focus on this side. Across traversals, the trend is quite similar, with a drop of true and detected matches at the edges of every traversal likely due to fewer reconstructed points to project.



(a)



(b)

Figure 6.3: Plots of the evolution of detected and "true" matches across traversals. Figure (a) reports data for the left camera, while (b) refers to the right camera. Pseudo-ground truth matches (blue line) show periodicity across traversals, and their number is smallest around the edges of each traversal. Detected matches are significantly fewer, and seem to vary with the date of the traversal. Sequences affected by match scarcity can be observed in the last segment of the left traversal (a), where the number of matches drops despite a still significant number of projected points.

An interesting overall growth in the number of detected matches is observed around the winter, i.e. in the dates from 2010-11-22 to 2011-03-04. We attribute this phenomenon to the largest similarity of these traversals to the database, which was shot on the 2011-04-04. We obtain visual confirmation of this hypothesis in Figure 6.4, which shows larger similarity to the database for the queries in late autumn up to early spring.

Figure 6.4: Display of corresponding images by date. It is observed that the most similar visual conditions to the reference image are dates from 2010-11-22 to 2011-03-04.

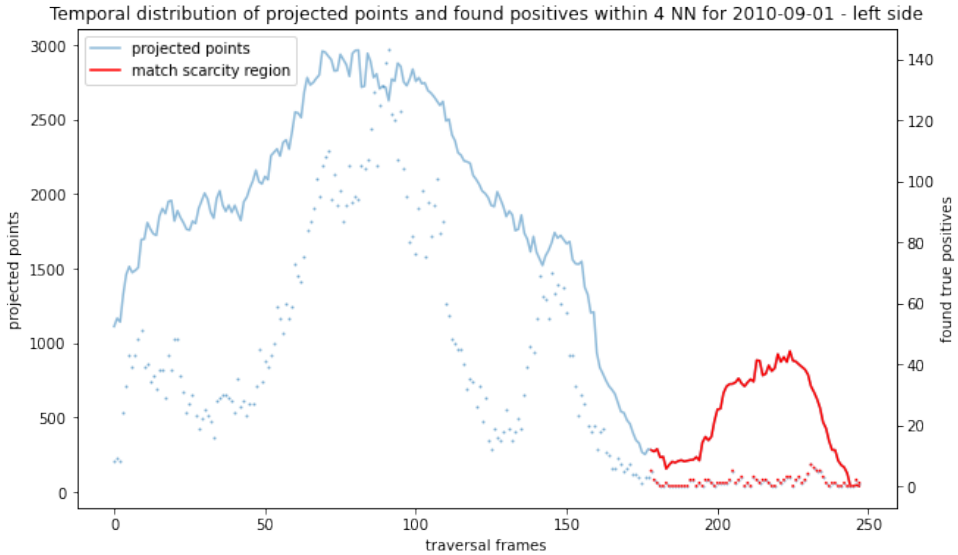


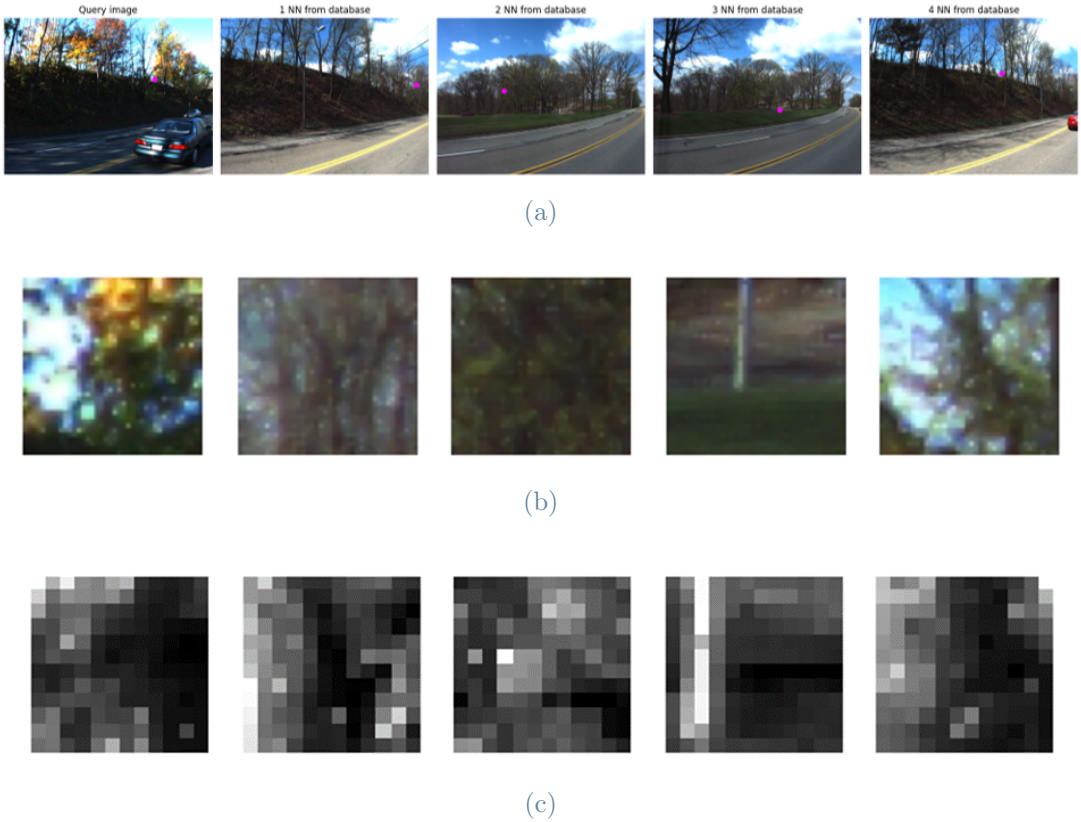
Figure 6.5: A close-up of the projected points and detected matches for the traversal of 2010-09-01, indexed by the frame order of the traversal. The highlighted area (in red) shows signs of match scarcity.

In the left traversal, a drop in the number of projected points is observed a few shots earlier than the end of the entire sequence, on all traversals. Concurrently, we find a drop in the number of detected positives, which remain quite low until the end of the sequence. For clarity, we highlight such area for the traversal of 2010-09-01 in Figure 6.5. We can observe how the number of detected matches oscillates but remains above 20 for most frames, until, around the 180th frame, the number of detected matches drops well below 10. We thus identify such region as appropriate for our purposes. Indeed, the drop in projected points is likely due to a sparser point cloud around the area, which in turn can be attributed to highly ambiguous scenes, or match scarcity at reconstruction time. Moreover, the lack of detected matches in the subsequent shots confirms that the problem of match scarcity is persistent in the area.

We select 70 images from these traversals for our experiments. To choose the most relevant images, we observe how many additional true matches are detected when passing from one to four nearest neighbors, and select a sequence with consistent gain. We exclude the traversals from 2010-11-22 to 2011-03-04, as in these the long-term match scarcity problem is mitigated by the similar appearance to the database condition. Finally, visual inspection ensures the choice is relevant. We report some of these images in figure 6.6, and we verify the presence of long term appearance variations that cause mismatches in Figure 6.7. For the selected query image, the correct match is only found as fourth neighbor, as shown in the first row (a). A close-up (b) on the described areas highlights variability due to different appearance, which is confirmed when looking at the actual described patches (c). Indeed, observing the patches only, the similarity among the correct pair is blurred in background noise, causing unrelated patches to be identified as more similar than the correct one. Only by checking a larger portion of the image we may recover an intuition of the correct keypoint to be associated. Therefore, the example confirms the need to include global information in the matching process.

Having selected the target sequence with match scarcity, we are also interested in assessing the pose estimation performance of our methods in sequences that exhibit milder problems related to long-term changes and repeated structures, so to isolate different phenomena. We select three more sequences, two from the Park setting and one from Urban setting. In the Park setting we additionally look for a sequence without long-term variations, but with ambiguity given by vegetation as the prevailing content. To this end, we select 150 random images from all the traversal length, and from the dates between 2010-11-22 and 2011-03-04. We furthermore look for a sequence with abundance of matches and some man-made ambiguities. This is easily found by checking peaks of detected matches in Figure 6.3a. Samples from this sequence are reported in Figure 6.8.

Figure 6.6: Some images from the sequence with match scarcity. The scenario is very challenging, including adverse illumination and changes of appearance due to seasonality and sunlight.



(a)

(b)

(c)

Figure 6.7: Example of incorrect match caused by long-term variations and descriptor ambiguity. The first row reports the full images with relative keypoints associated. The leftmost image is the query, followed by the database images of the associated descriptors in the order of distance from the query descriptor. The correct keypoint is only selected as fourth-nearest neighbor. The second row is a closeup on the described area, while the third row reports the actual described patches.

Figure 6.8: Images containing repeated structures in the Park setting. Lines on the road and light poles represent recurring structures whose description becomes ambiguous (see Figure 6.10b).

Figure 6.9: Images containing repeated structures in the Urban. The recurrent architectural elements of the depicted building bring significant confusion at matching time (see Figure 6.10c).

Finally, in the Urban setting we select a sequence with repeated structures, to observe the methods behavior in ambiguous settings without long-term variations nor vegetation. Images from this sequence are reported in Figure 6.9.

To summarize, we will use four sequences in our analyses:

- **S1**: a hard sequence from Park with long-term variations, severe match scarcity and ambiguity of descriptors,
- **S2**: a medium sequence from Park without long-term variations but with prevailing vegetation,
- **S3**: an easy sequence from Park with abundance of matches and both man-made repeated structures and vegetation,
- **S4**: a Urban sequence with man-made repeated structures.

We seek visual confirmation of the ability to reach new valid matches in these sequences by searching neighbors beyond the first. Some examples from S1, S3 and S4 are displayed

in Figure 6.10.

6.2.4. Analysis of matching

The first experiment is a comparison of the matching performance of our algorithm and the most common matching techniques in literature. Methods with a focus on long-term settings such as Geometric-Semantic Match Consistency [22, 44], employ nearest neighbor matching with Lowe's ratio test and threshold at 0.9. The Simple Semantic Match Consistency method [27] also considers the first neighbor matches, but applies a semantic filter, discarding points with inconsistent semantic. Finally, the most efficient approaches such as Active Search [34] perform a strict filtering of nearest neighbor matches with a ratio test and threshold at 0.7. We compare our matching strategies to these options, and additionally plot the figures of a K-nearest neighbors matching with no filtering – this represents an optimistic boundary in terms of recall, although remaining impractical for its extremely low precision.
We start by analysing the achieved recall of the above listed filtering methods against the optimistic k-NN matching. We experiment on the Park sequence with match scarcity (S1). Figure 6.11 illustrates its trend, on average, as a function of the number k of explored neighbors. It is immediately evident how several matches are lost through filtering even with 1-NN only, and such difference is only marginally reduced, if not worsened, by the increasing number of considered neighbors and a more relaxed filter – indeed, we follow [49] and perform k-NN ratio test with the k+1th neighbor for all k neighbors.
The dashed lines further allows comparison with literature methods, which only consider the first nearest neighbor. The emerging fork of recall is quite significant. We note that, as k increases, the k-NN method can find nearly three-fold the amount of correct matches – compared to twice the amount for the first neighbor only – as the 1-NN with ratio test and threshold at 0.9. This evidence confirms there is room for large gains through considering additional neighbors.

We repeat the analysis including variations of our method. In particular, we assess the following design choices:

- inclusion of both semantic inliers and outliers in the semantic score. We indicate with *sum* the original version of [44], which does not account for the number of incorrectly projected points, and with *ratio* the option weighting the number of semantic inliers on the total number of projected points
- match selection with the score as measure of quality. We compare the proposed method, which only picks one match among the candidate k nearest neighbors

(a)

(b)

(c)

Figure 6.10: Plot of one example of incorrect first neighbor matching which is solved at the second neighbor. The sequences they refer to are respectively (a) S1, (b) S3 and (c) S4. By considering priority based on scores, we are able to retrieve the correct match in all cases.

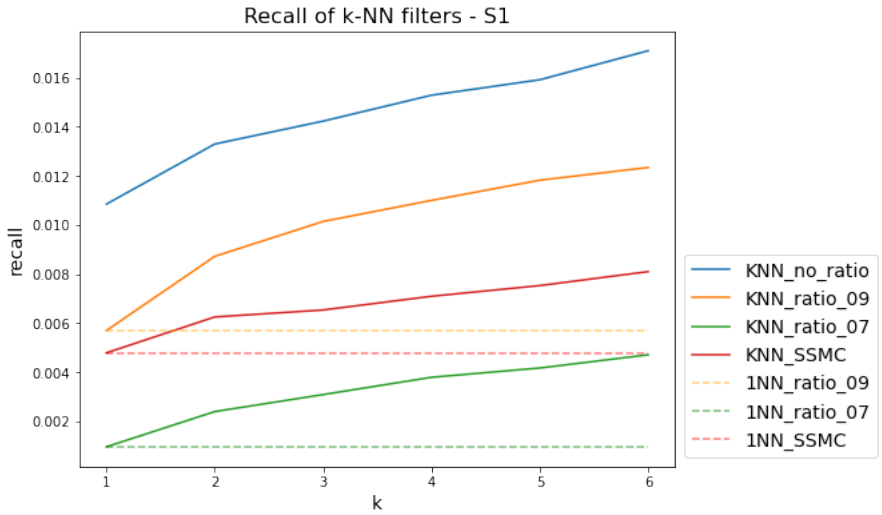


Figure 6.11: Plot of the recall of k-NN filters. Continuous lines consider a growing set of k neighbors at matching time, and assess the impact of filtering strategies only. Dashed lines represent methods from literature, with fixed set of matches at first neighbor only.

(*largest*), with keeping all matches whose score is above a fixed threshold (*all*)

- visibility filter for the score computation. We experiment with an alternative to the approach proposed by [44], where points to project are selected through their distance from the 3D point of the match, and the consistency of direction and distance from the estimated centre. As this approach forces the recalculation of the set of projected points for every pose hypothesis and hence increases the computational overhead, we test a faster version based on *covisibility* (cf. [35]). We project all points that are covisible with the 3D point we are matching to. We experimentally assess that projected points increase roughly of one magnitude order, thus enabling to exploit further the potential of fine-grained segmentation

Figure 6.12 summarizes the outcome of the experiment on S1. We observe an improvement on both recall and precision, with the exception of the 1-NN with ratio threshold of 0.7, which is extremely conservative and reaches a very high precision but also achieves too little recall. The exploration of more than one neighbor seems beneficial for recall, especially when passing from one to two nearest neighbors. The analysis shows a similar trend for all our methods, with the methods considering all neighbors achieving a slightly higher recall, and the covisibility filter locating in the middle. Methods employing the ratio strategy have lower precision, perhaps due to the low threshold on these scores, which was set to 0.001.

Overall, the analysed data is strongly in favor of our multi-neighbor matching strategy in

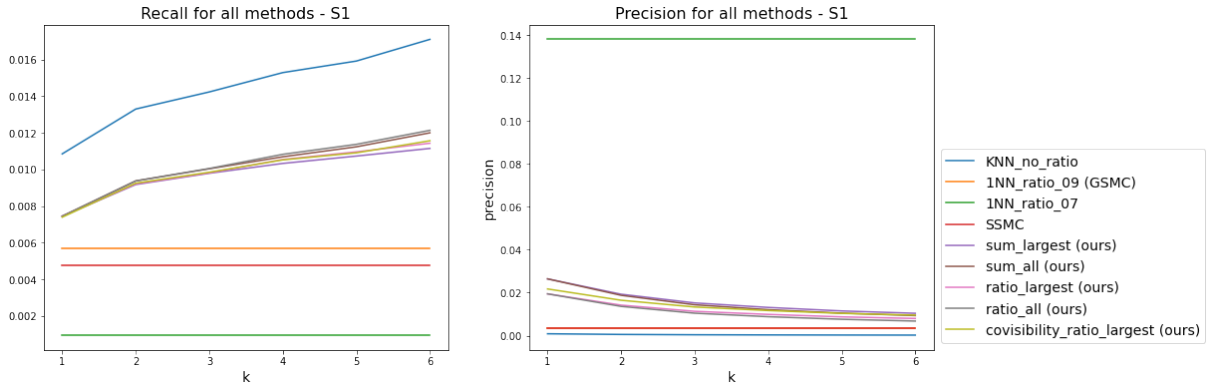


Figure 6.12: Plot of the recall (left) and precision (right) of all considered methods. Our methods span several design options for the semantic score. These include *ratio* or *sum*, which either compute the semantic measure as the ratio of semantic inliers on all projected points or as semantic inliers only as in the original version of [44], *largest* or *all*, which consider respectively the match with largest score among k-NN or all matches with score above some threshold, and *covisibility ratio*, which uses a covisibility-based filter to select points to project in the semantic score computation.

presence of match scarcity, as both recall and precision keep improving without saturating with higher k. This also confirms the shortcoming of filtering matches in disregard of the broader context, as both the ratio test and simple semantic match consistency filter give up on larger portions of matches than our context-based Semantic Matching even when matching with $k = 1$.

6.2.5. Study of model variations

We test in this section the impact on pose estimation of varying the following: (i) the use of one or k nearest neighbors, (ii) the width of the semantic check we perform before computing the score (see the implementation details 6.1.2), (iii) whether to keep the best among the k nearest matches or all those with score above some threshold, (iv) the use of visibility or covisibility as a filter for points in the score. Tests are carried out with $k = 2$, and we experiment with checking the semantic consistency of the 3D match with 5×5- and 15×15-large square windows of pixels around the 2D keypoint. We report in table 6.1 the results of pose estimates evaluation.

Percentages of correct localizations, S1

experiment type	covisibility	**Fine** 0.25m / 2°	**Medium** 0.5m / 5°	**Coarse** 5m / 10°
1-NN, 15×15		2.9	2.9	10.0
k-NN, largest, 5×5		2.9	2.9	8.6
k-NN, all, 5×5		0.0	1.4	5.7
k-NN, largest, 15×15		**7.1**	**8.6**	**17.1**
k-NN, all, 15×15		0.0	2.9	8.6
1-NN, 15×15	✓	2.9	2.9	4.3
k-NN, largest, 5×5	✓	2.9	4.3	5.7
k-NN, all, 5×5	✓	0.0	2.9	4.3
k-NN, largest, 15×15	✓	1.4	4.3	5.7
k-NN, all, 15×15	✓	1.4	4.3	12.9

Table 6.1: Pose estimation results for model variations in the Park sequence with model scarcity (S1).

The results show a clear dominance of the strategy employing k nearest neighbors, retaining the match with largest score only and checking semantic consistency on a 15×15-large window. Covisibility is observed to worsen localization performances, perhaps due to the lack of any control on the match visibility.

Similar conclusions emerge in the easiest Park sequence (S3). We report in table 6.2 the results, for the visibility configurations only. The overall very high localization percentages make it evident that the sequence is easier than the previous one. Moreover, the additional spanning of k nearest neighbors brings only marginal, although consistent, improvement on matching on the first nearest neighbor only, while the choice of one neighbor rather than keeping multiple ones appears quite important. We attribute this behavior to the wider amount of available inliers for model fitting, where correct additional matches would only refine the pose estimate, but a large amount of incorrect ones might prevent from sampling the correct model.

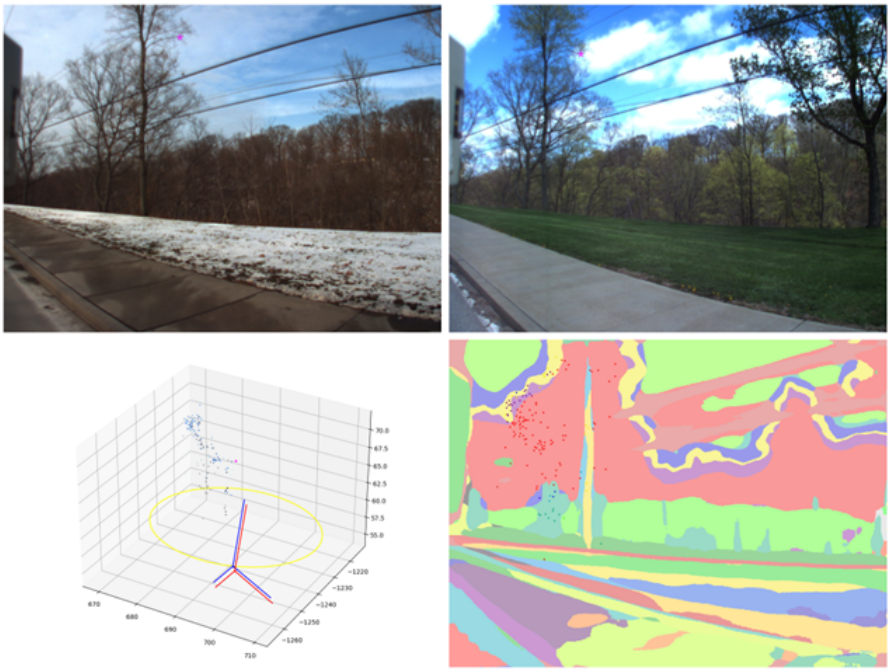
Percentages of correct localizations, S3

experiment type	covisibility	**Fine** 0.25m / 2°	**Medium** 0.5m / 5°	**Coarse** 5m / 10°
1-NN, 15 × 15		59.1	72.7	93.6
k-NN, largest, 5 × 5		59.1	70.9	**95.5**
k-NN, all, 5 × 5		33.6	59.9	84.5
k-NN, largest, 15 × 15		**60.0**	**73.6**	92.7
k-NN, all, 15 × 15		34.5	52.7	92.7

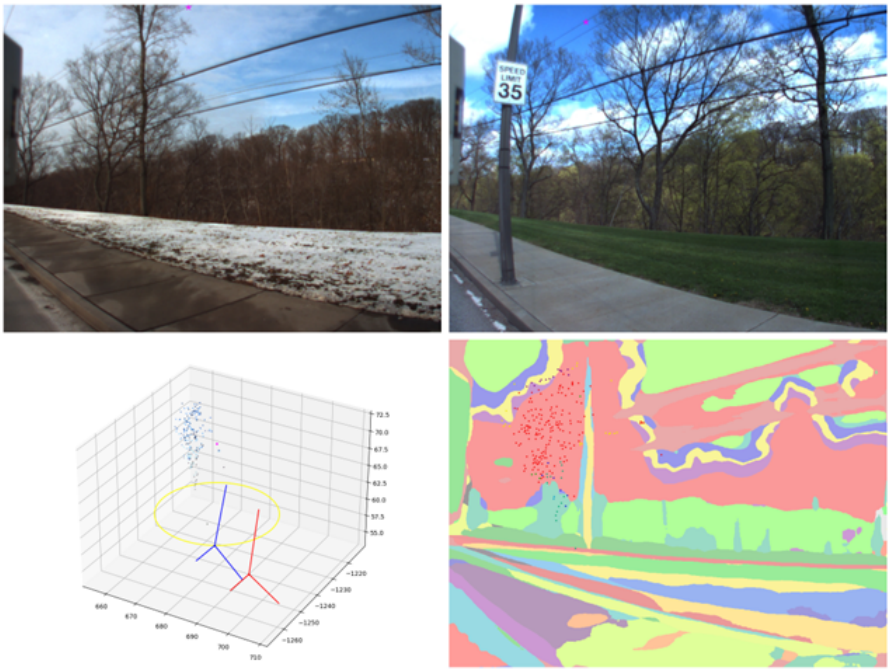
Table 6.2: Pose estimation results for model variations in the Park sequence with repeated structures (S3).

Another important design choice regards the score computation. In contrast to the original score from [44], we find it is important to weigh the score to the total amount of projected points, so to penalize scores that miss the projection of several inliers.

For an example, consider Figure 6.13, which reports two high scores associated to matches in the same query image from the Park slice of the CMU Seasons dataset. Top rows in the pictures depict the matched keypoints in their original image, with the query image on the left and its database correspondent on the right (note that we can identify a database image for the matched 3D point since there is a one-to-one relation between database descriptors we match to and database images, i.e. we search for nearest neighbors using all descriptors from the reconstruction). At the bottom of the pictures, a representation of the semantic projections on the query semantic mask is shown (right), and a qualitative comparison of the predicted and ground truth camera reference is given (left). The projected points are plotted in 3 dimensions, with the matched keypoint highlighted in fuchsia, the real camera reference in red, and the predicted reference in blue, with explored locations in yellow. Analyzing the poses, we can observe that only the first match (Figure 6.13a) is consistent with the camera pose, while the other is a mismatch (Figure 6.13b). Interestingly, when comparing these two cases of a success and a failure of the score, the correct match has fewer correct projected points –118 out of 151, compared to 175 out of 233 of the wrong match– since fewer points are projected overall. The failure seems to occur because a large portion of the scene is dominated by the red semantic class, and therefore a high number of projected points from that class drive the majority of the score.
We test the two options directly on pose estimation in table 6.3. Performances are overall



(a) Success case



(b) Failure case

Figure 6.13: Example of a (a) success and (b) failure case of the fine-grained semantic projection. The top row shows the correspondence in the query image and database image. The bottom right picture represents the semantic projection of the point cloud. The bottom left picture is a plot of the location of the estimated camera pose with respect to the 3D scene. We plot: the 3D point that was matched to a query point (fuchsia), the subset of points which were projected (with their RGB color), the centre locations spanned to compute the semantic score (yellow), the estimated camera axes (blue), the ground truth axes (red). Image (a) is observed to give a correct pose, while in image (b)- it is incorrect, despite the high number of semantic inliers.

better with the ratio strategy.

Percentages of correct localizations, S1

	Fine	Medium	Coarse
experiment type	0.25m / 2°	0.5m / 5°	5m / 10°
k-NN, largest sum	4.3	**10.0**	11.4
k-NN, largest ratio	**7.1**	8.6	**17.1**

Table 6.3: Pose estimation results for sum and ratio score versions in the Park sequence with model scarcity.

We finally validate the additional visibility check we add while computing the semantic score, whose benefits on matching are computational savings and increased precision. As we can see in table 6.4, the figures without visibility check, which retain more matches, are overall worse than the ones achieved with the check. We will thus use this strategy even in the computation of results of literature methods, so to yield a fair comparison of the matching strategy.

Percentages of correct localizations, S1

		Fine	Medium	Coarse
experiment type	visibility check	0.25m / 2°	0.5m / 5°	5m / 10°
1-NN, 15×15	✓	2.9	2.9	10.0
k-NN, largest, 5×5	✓	2.9	2.9	8.6
k-NN, all, 5×5	✓	0.0	1.4	5.7
k-NN, largest, 15×15	✓	**7.1**	**8.6**	**17.1**
k-NN, all, 15×15	✓	0.0	2.9	8.6
1-NN, 15×15		2.9	4.3	4.3
k-NN, largest, 5×5		2.9	4.3	8.6
k-NN, all, 5×5		0.0	1.4	4.3
k-NN, largest, 15×15		1.4	4.3	4.3
k-NN, all, 15×15		0.0	4.3	11.4

Table 6.4: Pose estimation results for visibility check variations in the Park sequence with model scarcity.

6.2.6. Pose estimation

Finally, we compare our method to existing algorithms in literature on the task of localization. Because no study includes results for the smaller sequences we choose to focus on, we reproduce all localization algorithms from the state of the art. In particular, our focus are the semantic-based algorithms presented in [22], hence Geometric-Semantic Match consistency (GSMC) and Simple Semantic Match Cosistency (SSMC). We furthermore add two baseline experiments whose matches are obtained with a ratio test (threshold 0.9) respectively on 1-NN and k-NN, and pose estimation is performed through RANSAC on the so obtained matches.

On our side, we evaluate three versions of localization algorithms. We create matches following the best performing matching strategy from the study of variations (see 6.2.5), which considers $k = 4$ nearest neighbors and retrieves the match with highest score: The score is computed with a semantic window of size 15×15 pixels, and the same visibility filter as [44] is used to select points to project in the score computation phase.

The first version we test consists on RANSAC pose estimation on the selected matches, with uniform sampling of models (*semantic matching - unbiased*). Scores are therefore only used to select matches and discarded afterwards.

A second version, directly inspired on the GSMC localization pipeline, employs scores during RANSAC model sampling. Scores are used as a measure of the match fitness to model estimation, with matches having higher scores being sampled with higher probability. For this method, we evaluate the version with and without Biased Consensus.

We summarize results of pose estimation on all four sequences in table 6.5.

Method / Setting m deg	Park hard(S1) 0.25/0.5/5 2/5/10	Park medium (S2) 0.25/0.5/5 2/5/10	Park easy (S3) 0.25/0.5/5 2/5/10	Urban (S4) 0.25/0.5/5 2/5/10
1-NN + ratio test 0.9 unbiased	0.0/0.0/0.0	30.7/42.0/49.3	32.7/36.4/44.5	70.0/75.5/88.2
k-NN + ratio test 0.9 unbiased	0.0/0.0/0.0	22.0/27.3/37.3	37.3/39.1/47.3	67.3/81.8/**95.5**
SSMC [22]	0.0/0.0/0.0	32.7/43.3/52.0	38.2/48.2/78.2	68.2/85.5/93.6
GSMC [22]	**2.9**/2.9/5.7	46.7/61.3/70.7	47.3/59.1/79.1	71.8/81.8/86.4
sem. matching unbiased	0.0/0.0/0.0	14.0/20.7/25.3	20.0/27.3/39.1	60.0/70.0/83.6
sem. matching biased sampling	**2.9**/4.3/8.6	**50.0**/**63.3**/**72.0**	**60.0**/**73.6**/**92.7**	**81.8**/**89.1**/91.8
sem. matching biased s. + c.	**2.9**/**7.1**/**20.0**	45.3/**63.3**/**72.0**	37.3/54.5/90.0	79.1/87.3/92.7

Table 6.5: Pose estimation results comparison on all sequences.

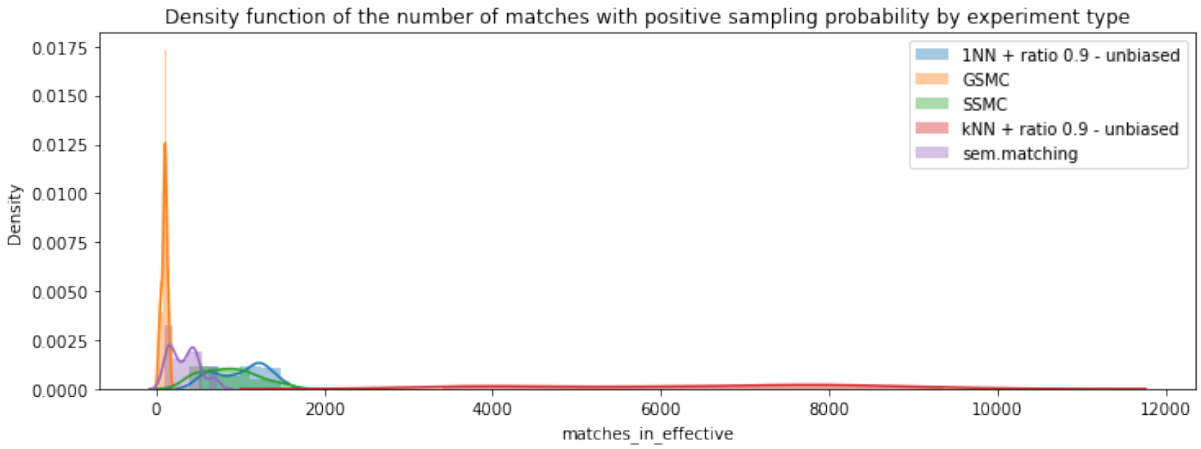
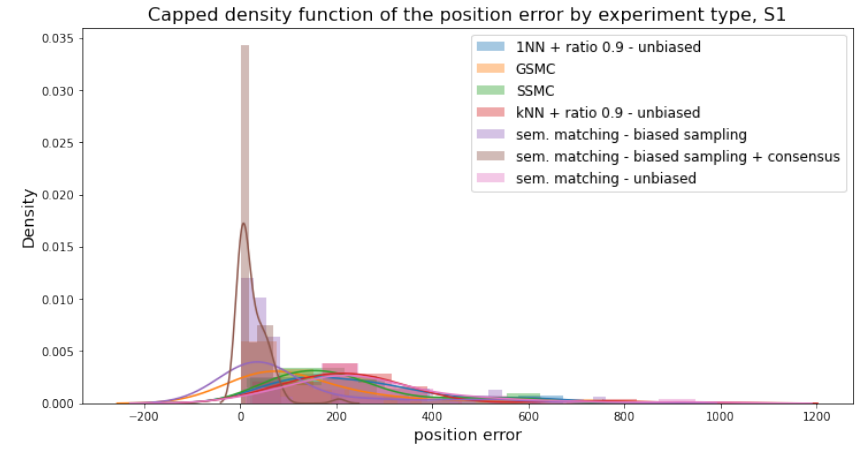


Figure 6.14: Distribution plot of the number of matches with positive sampling probabilities that are fed into RANSAC pose estimation for every algorithm.

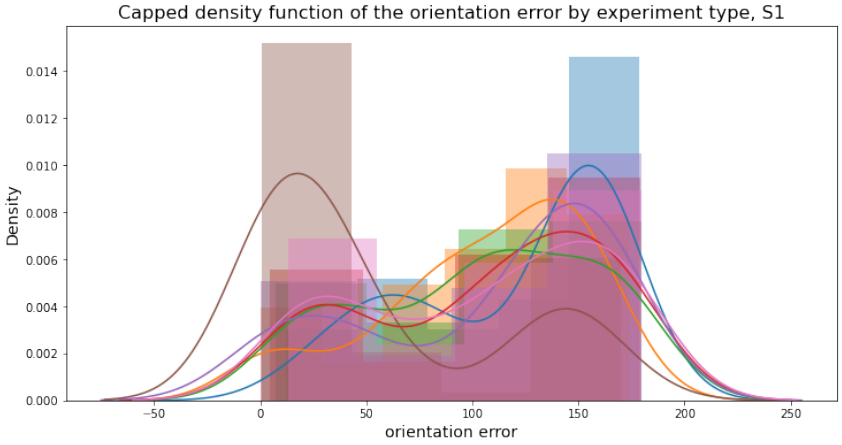
Localization results confirm the validity of our matching strategy, which proves decisive in all sequences for most accuracy levels. Moreover, the overall results in the Park sequence with match scarcity (S1) reaffirm the sequence is particularly hard, with several methods missing all localizations. Conversely, the other sequences gain overall high success, especially in the coarse localization level. The presence of vegetation always requires help of semantics and biased sampling, as we may see in all Park sequences. Instead, methods without semantics do not seem to be excessively penalized in urban scenarios.

Focusing on S1, we analyze further evidence to explain results more in depth.
Figure 6.14 reports the distribution of the number of matches with positive sampling probabilities used by each pose estimation algorithm throughout the sequence. It is worth noting that the Geometric-Semantic Match Consistency algorithm has the least amount of matches. Although yielding some correct localizations, this method is thus excessively reliant on the goodness of the filtered matches, which is not guaranteed with long-term variations as previously seen. On the other end of the spectrum, the methods that do not apply any filtering, have an excessively large pool of matches to sample from. Because of the low ratio of inliers to outliers, the correct model is never found. Our method collocates in between these extrema and can thus benefit from larger amounts of matches but still reasonably few models to explore to sample the correct model with high probability.

Another interesting highlight of the pose estimation experiment on S1 is the success of Biased Consensus. It increases the amount of correct coarse localizations of more than 10% from the baseline biased estimation method of [44]. The distribution plot of the pose and estimation errors for all models support this result (see Figure 6.15). We note how the pose error for the semantic biased sampling and consensus is much more likely to



(a)

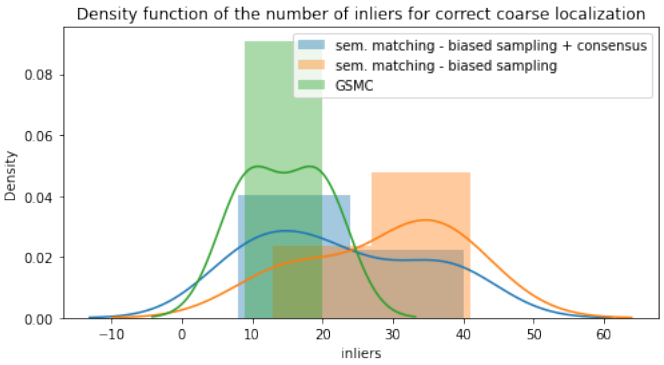


(b)

Figure 6.15: Plots of the distributions of (a) position and (b) orientation error on the Park sequence S1.

yield smaller errors, whereas other methods have more spread distributions, with mode on larger position and orientation errors.

This evidence clearly motivates the rationale behind Biased Consensus. In a context of match scarcity, the correct model should not be evaluated uniquely on the cardinality of its consensus set, since the probability that other models reach equal or higher consensus is large. This fact was noted before in [49], in presence of repeated structures.
By weighting the consensus with semantics, we increment the probability of the correct model to prevail, yet sometimes penalizing the absolute number of inliers. Indeed, as shown in Figure 6.16, fewer inliers are found than without biased consensus.



(a)

Figure 6.16: Plots of the distributions of inliers on the Park sequence with match scarcity. The plot only considers queries that were correctly localized, at least at coarse level.

7 | Conclusions and future developments

In this section we summarize the findings of this work and explore promising research directions.

7.1. Findings

In this work we have presented a novel matching strategy for long-term visual localization with severe match scarcity.

We have observed the similarities between the long-term setting and scenes dominated by man-made repeated structures –i.e. repetitive patterns appearing across images– from which we have learned the importance of global information to solve local ambiguities directly at matching stage. We have noted how appearance variations cause some correct matches to fall beyond the nearest neighbor, but several of these remain within the first few neighbors.

We have found that previous traditional localization methods aimed at long-term visual localization are unsuited to localize with severe match scarcity, because their aggressive filtering can often discard matches valuable for pose estimation, worsening the scarcity problem. While reaffirming the validity of using semantics as orthogonal information to pinpoint correct matches from a large set of incorrect ones, we have demonstrated the benefits of acting at an earlier stage than other methods, that is before forming correspondences. Through ad-hoc experiments, we have shown our method can increase the amount of retained correct matches (recall), while also improving the purity of the set of matches (precision). When testing directly on pose estimation, our matching strategy enabled us to achieve the highest percentage of correct localization in most precision levels of all tested sequences.

Finally, we have proposed a variation of RANSAC pose estimation algorithm which employs semantics extensively, not only to sample models but also to evaluate consensus.

The method has shown improved localization abilities of over 14% at coarse level on the biased sampling method of [44], for a hard sequence of images with match scarcity.

7.2. Future work

7.2.1. Matching with visual attention masks

As emerges from our work, matching in long term scenarios is a key area to be improved for robust localization.
The success of learning based methods, and especially HF-Net [31] (cf. Section 4.6.1), shows that global information is crucial even when establishing local correspondences, since it solves ambiguities and brings up matches that would not be formed in the first place. Our approach uses the semantic content of the whole scene to inject context into the matching process, and we demonstrated its effectiveness throughout the whole localization pipeline.
As a future research direction, it would be interesting to explore alternative matching strategies in this sense. Among these, we have particularly looked at visual attention, a mechanism which learns to focus on parts of an image or scene, often in an input dependent manner, while learning to perform another task.

According to [14], computer vision tasks mainly employ two types of attention: *channel* and *spatial*. The first helps focusing on different concepts, as it places weights on the network channels – which stand to represent concepts learned from the database. The second allows to highlight spatial locations in the picture.
Channel and spatial attention have been successfully included in many state-of-the-art methods for the tasks of place recognition and instance retrieval. Similar to visual localization, in these tasks we look for a place within a database, but rather than outputting a precise pose, we can consider a successful localization if we are able to identify the rough location of the query, in the form of GPS coordinates, landmark tags or similar images.

Place recognition and image retrieval tasks are well suited to end-to-end learning. In fact, these can be naturally cast as classification or ranking tasks, and a neural network can be trained to them provided that a general enough database is collected. As in many other vision tasks, the power of learning has soon made CNNs the standard for coarse localization.
Two families of approaches are mainly adopted. The first consists in training a global descriptor of the image and using it to perform the end task. Spatial attention has been used to improve the impact of local features on the global representation, both in a learned

[19, 48] and non-learned way [18, 30, 45]. A second group of methods, instead, defines the similarity of images directly on local features, to improve robustness to occlusion and clutter. These finer-grained spatial descriptions of the images can be obtained effortlessly in the same pipeline as used in the first family of methods, by simply extracting the features before pooling them into a global descriptor. Attention can then be used to discard the least interesting local features, only retaining necessary information [29, 46]. Some further research has directed towards integrating these two forms of description [5], or combining spatial and channel attention [41].

When it comes to pose estimation, learning-based methods are still lagging behind traditional methods [38]. Perhaps for this reason, the potential of visual attention for visual localization remains largely unexplored.
In light of the several common points among coarse and fine localization, we ask the following: *Are attention maps indicative of the usefulness of regions or objects in the long-term visual localization task?* Future work would then aim at answering the above question by experimenting with attention trained on retrieval tasks, for instance in the global descriptor head of the HF-Net architecture.

7.2.2. Estimating a pose from semantic scores

The semantic score we have used throughout this work carries much more information than an individual match does. We have often underlined its ability to measure the consistence of the putative match to the broader context. One additional valuable data point we can obtain is a rough estimate of the camera pose, as product of the geometry of the match and the exploration of the remaining pose hypotheses through semantic projection.
These poses are virtually valid estimates for the camera pose, although they might be quite noisy. It would be interesting to understand whether (*i*) these poses can lead to reaching the largest consensus set faster than RANSAC – or its improved versions such as PROSAC [8] – if used in place of randomly sampled P3P models, (*ii*) the distribution of poses from all matches can be used to infer the location of the real pose consistently. Regarding the latter question, we recall that poses lay in 6-dimensional manifolds, whose dimensions we can further reduce through the additional information required by the score (known gravity direction and camera height). One should therefore study the distribution of scores in this context. One possible instrument to do it is the mean shift clustering algorithm [7], designed to find modes of a distribution. Each of those could be tested on consensus, yielding a faster estimate than through RANSAC.

Bibliography

[1] R. Arandjelović and A. Zisserman. Visual vocabulary with a semantic twist. In *ACCV*, 2014.

[2] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2911–2918, 2012. doi: 10.1109/CVPR.2012.6248018.

[3] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition, 2015. URL `https://arxiv.org/abs/1511.07247`.

[4] H. Badino, D. Huber, and T. Kanade. The CMU Visual Localization Data Set. `http://3dvis.ri.cmu.edu/data-sets/localization`, 2011.

[5] B. Cao, A. Araujo, and J. Sim. Unifying deep local and global features for image search, 2020. URL `https://arxiv.org/abs/2001.05027`.

[6] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features, 2018. URL `https://arxiv.org/abs/1807.05520`.

[7] Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995. doi: 10.1109/34.400568.

[8] O. Chum and J. Matas. Matching with prosac - progressive sample consensus. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 220–226 vol. 1, 2005. doi: 10.1109/CVPR.2005.221.

[9] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding, 2016. URL `https://arxiv.org/abs/1604.01685`.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[11] D. DeTone, T. Malisiewicz, and A. Rabinovich. Superpoint: Self-supervised interest point detection and description. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 337–33712, 2018.

[12] O. Enqvist, E. Ask, F. Kahl, and K. Åström. Robust fitting for multiple view geometry. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Computer Vision – ECCV 2012*, pages 738–751, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-33718-5.

[13] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, jun 1981. ISSN 0001-0782. doi: 10.1145/358669.358692. URL `https://doi.org/10.1145/358669.358692`.

[14] M. Guo, T. Xu, J. Liu, Z. Liu, P. Jiang, T. Mu, S. Zhang, R. R. Martin, M. Cheng, and S. Hu. Attention mechanisms in computer vision: A survey. *CoRR*, abs/2111.07624, 2021. URL `https://arxiv.org/abs/2111.07624`.

[15] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2004. doi: 10.1017/CBO9780511811685.

[16] O. Herranen. *Social Institutions and the Problem of Order: A Relational Approach to Neo-Institutionalism through Social System Theory, Social Constructionism, and Critical Ideology Theory*. PhD thesis, 01 2020.

[17] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3304–3311, 2010. doi: 10.1109/CVPR.2010.5540039.

[18] Y. Kalantidis, C. Mellina, and S. Osindero. Cross-dimensional weighting for aggregated deep convolutional features. *CoRR*, abs/1512.04065, 2015. URL `http://arxiv.org/abs/1512.04065`.

[19] H. J. Kim, E. Dunn, and J.-M. Frahm. Learned contextual feature reweighting for image geo-localization. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3251–3260, 2017. doi: 10.1109/CVPR.2017.346.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran As-

sociates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[21] Z. Kukelova, M. Bujnak, and T. Pajdla. Closed-form solutions to minimal absolute pose problems with known vertical direction. In *ACCV*, 2010.

[22] M. Larsson, E. Stenborg, C. Toft, L. Hammarstrand, T. Sattler, and F. Kahl. Fine-grained segmentation networks: Self-supervised segmentation for improved long-term visual localization. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 31–41, 2019.

[23] Y. Li, N. Snavely, and D. P. Huttenlocher. Location recognition using prioritized feature matching. In *Proceedings of the 11th European Conference on Computer Vision: Part II*, ECCV'10, page 791–804, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3642155510.

[24] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, nov 2004. ISSN 0920-5691. doi: 10.1023/B:VISI.0000029664.99615.94. URL https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[25] W. Maddern, G. Pascoe, C. Linegar, and P. Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1): 3–15, 2017.

[26] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP*, 2009.

[27] M. Måns Larsson, E. Stenborg, L. Hammarstrand, M. Pollefeys, T. Sattler, and F. Kahl. A cross-season correspondence dataset for robust semantic segmentation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9524–9534, 2019. doi: 10.1109/CVPR.2019.00976.

[28] G. Neuhold, T. Ollmann, S. R. Bulò, and P. Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5000–5009, 2017.

[29] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-scale image retrieval with attentive deep local features, 2016. URL https://arxiv.org/abs/1612.06321.

[30] F. Radenović, G. Tolias, and O. Chum. Fine-tuning cnn image retrieval with no human annotation, 2017. URL https://arxiv.org/abs/1711.02512.

[31] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk. From coarse to fine: Robust hierarchical localization at large scale, 2018. URL `https://arxiv.org/abs/1812.03506`.

[32] T. Sattler. Feature-based visual localization. presentation, Czech Institute of Informatics, Robotics and Cybernetics Czech Technical University in Prague, 2021.

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

[34] T. Sattler, B. Leibe, and L. P. Kobbelt. Improving image-based localization by active correspondence search. In *ECCV*, 2012.

[35] T. Sattler, M. Havlena, F. Radenović, K. Schindler, and M. Pollefeys. Hyperpoints and fine vocabularies for large-scale location recognition. 12 2015. doi: 10.1109/ICCV.2015.243.

[36] T. Sattler, B. Leibe, and L. P. Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1744–1756, 2017.

[37] 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. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[38] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[39] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016. doi: 10.1109/CVPR.2016.445.

[40] Sivic and Zisserman. Video google: a text retrieval approach to object matching in videos. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 1470–1477 vol.2, 2003. doi: 10.1109/ICCV.2003.1238663.

[41] C. H. Song, H. J. Han, and Y. Avrithis. All the attention you need: Global-local, spatial-channel attention for image retrieval. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 439–448, 2022. doi: 10.1109/WACV51458.2022.00051.

[42] L. Svärm, O. Enqvist, M. Oskarsson, and F. Kahl. Accurate localization and pose estimation for large 3d models. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 532–539, 2014. doi: 10.1109/CVPR.2014.75.

[43] C. Toft, C. Olsson, and F. Kahl. Long-term 3d localization and pose from semantic labellings. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 650–659, 2017. doi: 10.1109/ICCVW.2017.83.

[44] C. Toft, E. Stenborg, L. Hammarstrand, L. Brynte, M. Pollefeys, T. Sattler, and F. Kahl. Semantic match consistency for long-term visual localization. In *ECCV*, 2018.

[45] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations, 2015. URL `https://arxiv.org/abs/1511.05879`.

[46] G. Tolias, T. Jenicek, and O. Chum. Learning and aggregating deep local descriptors for instance-level recognition, 2020. URL `https://arxiv.org/abs/2007.13172`.

[47] A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:257–271, 2018.

[48] M. Yang, D. He, M. Fan, B. Shi, X. Xue, F. Li, E. Ding, and J. Huang. DOLG: single-stage image retrieval with deep orthogonal fusion of local and global features. *CoRR*, abs/2108.02927, 2021. URL `https://arxiv.org/abs/2108.02927`.

[49] B. Zeisl, T. Sattler, and M. Pollefeys. Camera pose voting for large-scale image-based localization. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2704–2712, 2015. doi: 10.1109/ICCV.2015.310.

[50] W. Zhang and J. Kosecka. Extraction, matching and pose recovery based on dominant rectangular structures. In *First IEEE International Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis, 2003. HLK 2003.*, pages 83–91, 2003. doi: 10.1109/HLK.2003.1240862.

[51] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6230–6239, 2017.

List of Symbols

List of variables

Variable	Description
$\boldsymbol{x}_i$	image keypoint coordinates, point in $\mathbb{R}^2$
$\boldsymbol{X}_j$	world object coordinates, point in $\mathbb{R}^3$
$\tilde{\boldsymbol{x}}$	homogeneous image coordinates, point in $\mathbb{P}^2$
$\tilde{\boldsymbol{X}}$	homogeneous world coordinates, point in $\mathbb{P}^3$
$(\boldsymbol{x}, \boldsymbol{X})$	2D-3D correspondence
I	RGB image in $\mathbb{R}^{H \times W \times 3}$
M	semantic mask of an image I, in $\{1, ..., L\}^{H \times W}$
P	camera matrix in $\mathbb{R}^{3 \times 4}$
R	rotation matrix in SO(3)
$\boldsymbol{t}$	translation vector in $\mathbb{R}^3$
$\boldsymbol{C}$	camera centre in $\mathbb{R}^3$
$\boldsymbol{f}$	descriptors of query keypoints, vectors in $\mathbb{R}^D$
$\boldsymbol{d}$	descriptors of database and point cloud keypoints, vectors in $\mathbb{R}^D$
d	distance of reconstructed points in the Euclidean space
p	outlier-ness probability of a match
m	minimal sample size for pose estimation
θ	RANSAC model

List of sets

Set	Description
$\mathcal{X}$	point cloud, set of points in $\mathbb{R}^3$
$\mathcal{M}$	set of matches in $\mathbb{R}^2 \times \mathbb{R}^3$
$\mathcal{Q}$	query keypoint set in $\mathbb{R}^2$
$\mathcal{I}$	image set, set of RGB images in $\mathbb{R}^{H \times W \times 3}$
$\mathcal{F}$	query descriptors set in $\mathbb{R}^D$
$\mathcal{P}$	point cloud descriptors set in $\mathbb{R}^D$

Acknowledgements

Vorrei ringraziare il Prof. Giacomo Boracchi, il Prof. Luca Magri e il Dr. Antonino Rizzo per la fiducia riposta in me per questo progetto di tesi e le tante opportunità di apprendimento che mi hanno offerto in questi sette mesi di lavoro. Sono stati, indubbiamente, mesi indimenticabili, in cui ho imparato ben più di una manciata di nozioni di computer vision. Se non altro, mi hanno trasmesso l'importanza di martellare, con pazienza e meticolosità.

Ringrazio poi i miei familiari e amici per il continuo sostegno durante il percorso di studi, nonostante le mie lunghe assenze. In particolare, ai miei genitori va la mia più profonda gratitudine per non avermi fatto mai mancare nulla e aver assecondato ogni mio sogno fin qui. Ai miei nonni vanno i ringraziamenti di chi ha ricevuto tanto da non potersi sdebitare. Infine, ai miei zii va la mia riconoscenza per avermi sempre pensato, pur da lontano, ed avermi consigliato nelle mie scelte. Spero di potervi rendere ogni giorno più orgogliosi.