Discriminative Feature-to-Point Matching in Image-Based Localization

Michael Donoser and Dieter Schmalstieg
Institute for Computer Graphics and Vision
Graz University of Technology
{donoser,schmalstieg}@icg.tugraz.at

Abstract

The prevalent approach to image-based localization is matching interest points detected in the query image to a sparse 3D point cloud representing the known world. The obtained correspondences are then used to recover a precise camera pose. The state-of-the-art in this field often ignores the availability of a set of 2D descriptors per 3D point, for example by representing each 3D point by only its centroid. In this paper we demonstrate that these sets contain useful information that can be exploited by formulating matching as a discriminative classification problem. Since memory demands and computational complexity are crucial in such a setup, we base our algorithm on the efficient and effective random fern principle. We propose an extension which projects features to fern-specific embedding spaces, which yields improved matching rates in short runtime. Experiments first show that our novel formulation provides improved matching performance in comparison to the standard nearest neighbor approach and that we outperform related randomization methods in our localization scenario.

1. Introduction

Image-based localization deals with the problem of how to precisely recover the 3D camera pose from a query image within a known 3D world. The prevalent approach in this field is to localize the query with respect to a 3D point cloud obtained by any Structure-from-Motion (SfM) pipeline. This has a wide range of applications such as the localization of user community photos [12, 15, 19] or the continuous localization in videos as used in augmented reality [1] or autonomous robotics applications [16].

The current state-of-the-art in image-based localization can mainly be divided into two different approaches: (a) image retrieval techniques (IR) and (b) brute-force nearest neighbor matching methods (BF). IR based methods first identify a subset of images of the database that are most similar to the query image, using effective techniques such as the vocabulary tree [17]. Afterwards, only 3D points visible within the first ranked images are considered for matching, which significantly reduces the number of matching candidates and consequently the runtime. On the positive side, such an approach scales well with the size of the 3D model. On the negative side, localization performance mainly depends on the availability of fairly similar images in the dataset, an assumption that is not necessarily valid for localization from community photos.

The BF localization strategy, which bypasses the image retrieval step by matching directly between the query features and the entire point cloud, consistently leads to improved localization quality, as is e. g. demonstrated by Sattler *et al*. [19]. In such a setup, matching speed and memory requirements are severely dependent on the overall number of descriptors assigned to the 3D points, since all the descriptors are considered for matching. In order to reduce the complexity the descriptors are summarized into the centroid per 3D point and a visual vocabulary strategy is again used for the nearest neighbor search.

In this paper, we follow this most promising strategy of brute-force 2D-to-3D matching and focus on scenarios where a rough position is provided by an external sensor like GPS. This allows to constrain the search space, which as a consequence reduces the number of 3D point candidates that have to be considered in matching. We present a method which (a) has a runtime independent of the actual 3D point cloud size and a manageable memory profile, (b) exploits all the information available from the 3D reconstruction and (c) provides accurate localization performance. We formulate the required matching as a discriminative classification problem and propose a novel random ensemble method designed for the requirements in such a scenario.

In general classification has some advantages for image-based localization in comparison to the standard nearest neighbor based localization. First, matching based on a classifier makes the runtime independent of the actual point cloud size, which allows to exactly predict the runtime to find correspondences, whereas in the BF approach runtime depends on the number of 3D points. Second, classification based matching exploits the set of 2D descriptors available per 3D point in a discriminative manner, which improves matching performance, as we also demonstrate in the experiments. Third, our classifier implicitly defines a probabilistic ranking of all correspondence hypotheses by analyzing the returned posterior probabilities. This ranking can be exploited in the subsequent hypothesize-and-verify framework. As shown in PROSAC [4], a reasonable ranking of the correspondence hypotheses significantly improves the runtime and sometimes even model fitting performance. Fourth, our method has a reasonable memory profile since we do not have to keep all point cloud descriptors in memory. Finally, our proposed approach is an anytime algorithm. Hence it is possible to stop it after only some of the base classifiers are evaluated, since a single base classifier is already sufficient to provide a first guess of the valid correspondences.

Note that our localization setting poses specific requirements on the classifier. First, we have to handle a large number of classes (number of 3D points in the sparse point cloud). Second, the number of training samples might differ a lot, since some 3D points might be represented by only a small (at least two) 2D descriptors, whereas others might have up to hundreds of samples. In general, the number of available training samples is in comparison to other fields quite low. Finally, computation time is also a limiting factor, since we want to apply the method in real-time fields like robotics or augmented reality.

For this reason, we base our approach on the random ferns classifier [18], which already has shown to be quite efficient and effective in other vision domains such as tracking, semantic segmentation and image classification. Nevertheless, as we show in the experiments a standard random fern classifier does not provide sufficient quality in our scenario, since it is not able to handle the small number of available training samples. In general, it is known that random ensemble methods only unfold their full potential if having access to a sufficient number of training samples [5].

In this paper, we propose an extension which considers discriminative embeddings. The core idea is to apply a supervised dimensionality reduction method in order to map randomly selected feature dimensions into a subspace, where optimal splits can be easily derived. Due to the subspace projections, we get a flat assignment structure, which ensures low computation time in both training (no evaluation of any split criteria) and testing, with an only slightly increased memory profile (for saving the projection matrices). Our embedded fern has several important properties: (a) it offers the same advantages as related randomization methods, such as inherent multi-class classification, possible handling of high-dimensional input data, robustness to label noise and high efficiency, (b) intuitive integration of a supervised dimensionality reduction method in a flat assignment scenario in order to improve the splits and (c) consequently improved classification results in image-based localization scenarios.

2. Related Work

We first discuss the state-of-the-art in both of our contribution fields: random ensemble based classification and image-based localization.

Random Ensemble Methods Random ensemble methods such as the popular random forest have proven to be an indispensable tool in computer vision, mainly because of their inherent multi-class capability, efficiency and competitive performance. Random forests consist of an ensemble of decision trees, where each tree splits the feature space in a hierarchical manner and class assignments are probabilistically inferred based on counting the number of class occurrences in the leaf nodes. Our method addresses a novel way to define the required splits. Such split variants, as well as the corresponding evaluation criteria were addressed in several recent research papers.

Gall et al. [7] considered vectors pointing from local object parts to the corresponding centroid for improving the splits in a random forest setting. They adapted the information gain criterion by either minimizing the class label uncertainty or the variability of the corresponding center votes per node. State-of-the-art results in object localization tasks were shown. Zhang *et al.* [24] address the issue of how to combine features obtained by diverging modalities into a random forest classifier. They used a mutual information criterion in order to select the best splits in each node. As specific application, photos were matched to sketches, which lead to promising results in this challenging field. Leistner *et al.* [13] extended the random forest framework in order to learn from weakly labeled videos. They adapted the node split criterion by adding a pair splitting error measure, analyzing patch pairs as labeled training data. The authors then showed improvements in applications such as object detection and tracking. Yao *et al.* [23] adapted a random forest for the task of fine-grained image categorization. In contrast to conventional decision trees, they used strong classifiers (support vector machines) at each node and combined information at different levels of the tree in order to build powerful trees, while maintaining low correlation between them.

Image-Based Classification As outlined in the introduction, brute-force 2D-to-3D [19] and image retrieval based [12] methods are considered to be state-of-the-art in the field of image-based localization. Recently some further extensions were proposed, mostly addressing complexity issues of these methods.

Li *et al*. [15] explicitly address the problem of localizing images within extremely large 3D point clouds of up to 70 million 3D points, focusing on scenarios with completely unknown position. The authors follow the direct 2D-to-3D matching approach [14, 19] and as contribution propose an improvement of the applied RANSAC scheme, by sampling the random hypotheses considering co-occurrence priors. Furthermore, they also analyze the possible performance gain by additionally applying an inverse matching from the 3D point cloud to 2D query features. Lim *et al*. [16] address large-scale problems by reducing the matching effort based on an initial clustering of the 3D point cloud into location classes. During localization the most likely class is selected and standard brute-force 2D-to-3D matching is applied for the constrained point cloud. The authors demonstrate accurate localization performance for micro aerial vehicles, achieving real-time performance by using Kalman filter based feature tracking.

Another branch of research focuses on improving efficiency by defining reasonable search strategies in order to identify the set of valid correspondences. An active search strategy was proposed in [20], where only a fraction of points has to be considered in order to infer the camera pose. The core idea is to define a priority queue for point matching, which exploits both appearance and co-visibility cues. Such a fast guided search was also proposed in [3], where a conditional probability of visibility is combined with a distance measure in order to prioritize specific points. In [2, 9] also discriminative learning is used in a localization scenario. However, both works aim at improving the similarity scores between Bag-of-Words image representations, whereas we use discriminative learning for finding 2D-to-3D matches between descriptors.

3. Localization as a Classification Problem

As a starting point, we assume that we have access to a sparse 3D point model, obtained by any Structure-From-Motion (SfM) pipeline such as Bundler [21]. Given a query image $\mathcal{Q}$, the goal of image-based localization is to identify the precise 6-DoF camera pose in relation to the provided sparse 3D point cloud. The main steps of such an approach are: (1) feature extraction in $\mathcal{Q}$, (2) establishing 2D-to-3D correspondences through feature matching and (3) estimating the camera pose by robustly solving an absolute pose problem in a hypothesize-and-verify loop.

We follow the principle of brute-force 2D-to-3D matching as proposed by Sattler *et al*. [19]. Our core idea is to replace the approximate nearest neighbor assignment by a discriminative classification step, essentially addressing step 2 of the aforementioned pipeline.

Our given sparse 3D point model consists of M 3D points P^j. For each 3D point P^j we have additionally given a list of descriptors $\mathbf{D}_i^j \in \mathbb{R}^D$ containing the feature descriptions of all the image points that were used to triangulate P^j. Note that some of the 3D points can have up to hundreds of associated feature descriptors, and that the points with the largest number of corresponding descriptions are presumably the most informative for localization, e. g. a point on a tower that can be seen from many viewpoints. Our goal is to match the N features $\mathbf{Q}_i \in \mathbb{R}^D$, detected in the query image $\mathcal{Q}$, to the M 3D points P^j. State-of-the-art in direct 2D-to-3D matching [19] reduces the list of descriptors per 3D point to a single one, and afterwards applies an approximate nearest neighbor match. In contrast, we aim to use all the available information by formulating matching as a classification problem. This allows to use all descriptors for training, since we do not have to keep the training samples after training the classifier. The basic idea is that each 3D point is represented by a unique class label $(1 \dots M)$, and that the corresponding descriptors are used as labeled training data for discriminative classification. Note that in this way, we exploit all descriptors in a discriminative setting (in contrast to a nearest neighbor assignment), which yields an improved set of correspondences.

After training we pass all the query descriptors $\mathbf{Q}_i$ to the classifier and obtain a $1 \times M$ vector of class-conditional probabilities representing certainty information about hypothesized correspondences to the 3D points. In order to remove unreliable assignments, we perform an additional verification step, conceptionally related to the widely applied ratio test in nearest neighbor (NN) matching. In NN matching a valid correspondence is only established if the distance ratio between the two nearest neighbors is smaller than a fixed threshold ϵ. Such an approach requires access to all descriptors during camera pose estimation. In contrast, our classification approach does not need to save the descriptors, since we fix the correspondence if the distance ratio between the highest and the second highest class-conditional probability is below ϵ.

For classification, we introduce an extension of the random fern principle exploiting all available descriptors in a discriminative manner, while maintaining almost the same runtime and memory profile, in the next section.

4. Embedded Random Ferns

Our novel classifier is based on the random fern [18]. We first outline the basic principle of random ferns, and then introduce our novel embedding variant.

Random Ferns A random fern addresses efficiency and memory issues of a random forest by a flat leaf node assignment instead of building a hierarchical structure. This flat assignment strategy leads to a runtime improvement and decreased memory demands while mostly maintaining classification accuracy. Similar to random forest the fern consists of an ensemble of base classifiers. In each base classifier the feature space is uniquely split into $L = 2^S$ bins, by making S binary decisions. The class occurrences per bin are counted during training, which defines class-specific probabilities. The base classifier decisions are then combined in a semi-naive Bayesian manner.

The goal of training is to define the S split parameters and to estimate the class conditional probabilities $P(F_m = l|c_i)$ for each bin l of the ferns F_m and classes c_i. We simplify our notation as

$$p_{l,c_i} = P(F_m = l \,|\, c_i) \quad , \tag{1}$$

where l denotes one of the $L = 2^S$ bins of the fern F_m and p_{l,c_i} is the probability that bin l votes for class c_i. Therefore, we can write all class conditional probabilities in an $F \times L \times C$ (number of ferns $\times$ number of bins $\times$ number of classes) matrix $\mathcal{F}$, with the convention that $\sum p_{l,c_i} = 1$. During training the probabilities are estimated by simply counting the number of labeled data-points entering the corresponding bin by

$$p_{l,c_i} = \frac{N_{l,c_i}}{N_{c_i}} \quad , \tag{2}$$

where N_{l,c_i} is the number of labeled training samples of class c_i that evaluates to fern bin l, and N_{c_i} is the total number of samples for class c_i.

Once the fern matrix $\mathcal{F}$ is estimated from the training data, a novel test data point is simply passed to all ferns F_m and the probabilities of the corresponding bins l, that the test point evaluates to, are combined in a semi-naive Bayesian manner leading to an assignment to a class c^* by

$$c^* = \arg\max_i \; P(\mathbf{x} \,|\, c_i) = \prod\nolimits_{m=1}^{F} p_{l,c_i} \, . \tag{3}$$

Embedded Ferns We build our method on the random fern principle. We assume that we have been given a set of N data points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_N\}$, in a D-dimensional feature space, i.e., $\mathbf{x}_i \in \mathbb{R}^D$. Stacking the data points together yields our $N \times D$ training data matrix $\mathbf{X} \in \mathcal{R}^{N \times D}$, where a single element x_{id} stands for the d-th feature of the i-th data point.

Similarly to conventional random ferns, we aim at assigning a test data point to one of the $L = 2^S$ bins per fern, by defining S binary splits for each fern. Therefore, our goal is to find a mapping of the D-dimensional data points to an S-dimensional binary assignment vector $\mathbf{b}^m \in \{0,1\}^S$ for the m-th fern, which uniquely separates the feature space and allows us to assign each data point to one of the $L = 2^S$ bins. This assignment has the goal of optimally discriminating between the classes.

In our matrix notation, conventional ferns obtain the binary assignment vector $\mathbf{b}^m$ by randomly choosing feature IDs and analyzing the corresponding data point entries by

$$\mathbf{b}^m = \mathbf{X}^m > 0 \;\; , \;\; \mathbf{X}^m = \{x_{id} \mid i \in \{1,...,N\} \; , d \in \mathcal{G}^m\} \tag{4}$$

where $\mathcal{G}^m = (r_1^m, r_2^m, \ldots, r_S^m)$ is a set of S randomly chosen feature IDs.

Our core idea is to select a larger number $B > S$ of features and afterwards project the B features of the data points to the final S-dimensional space. For each fern we randomly select a fixed number of features $B \leq D$ but $B > S$, which reduces the input training matrix $\mathbf{X}$ to a fern specific $N \times B$ input matrix $\mathbf{X}^m = \{x_{id} \,|\, \forall\, i \text{ and } d \in \mathcal{G}^m\}$ for the m-th fern, where $\mathcal{G}^m$ this time is a set of B randomly chosen feature IDs. We further define the binary assignment vector $\mathbf{b}^m$ as a linear combination by

$$\mathbf{b}^m = (\mathbf{X}^m \, \mathbf{W}^m + \mathbf{o}^m) > 0 \;\; , \tag{5}$$

where $\mathbf{W}^m$ is a $B \times S$ projection matrix and $\mathbf{o}^m$ is a $S \times 1$ offset vector. The most straight forward way to define the projection matrix $\mathbf{W}^m$ is to choose it in a random manner, which yields a classifier frequently referred to as oblique forests [10] within the scope of decision trees. However, this would neglect the available training data labels, which can be exploited to find an improved projection. For this reason, we identify one projection matrix per fern which optimally discriminates the different classes from each other.

The core idea of our method is to use a supervised dimensionality reduction method in order to capture the structure of the training dataset $\mathbf{X}^m$ (consisting of the randomly selected features), and to identify optimal binary split parameters $(\mathbf{W}^m, \mathbf{o}^m)$. In order to do this, we propose to use Canonical Correlation Analysis (CCA) [11], a robust subspace method which effectively identifies a common latent space from two different views. In this way, we correlate the provided features and the corresponding label space, which constitutes the basis for finding optimal binary splits.

Assuming that each data point $\mathbf{x}_i$ has associated the corresponding label $y_i \in \{1, 2 \ldots C\}$, we can build a dataset label matrix $\mathbf{Y} \in \{0,1\}^{N \times C}$, where the corresponding entry y_{il} is 1 if data-point i belongs to class c and 0 otherwise. Canonical Correlation Analysis (CCA) tries to find two sets of basis vectors $\mathbf{w}_x$ and $\mathbf{w}_y$, so that the correlation between the projections $\mathbf{X}^m \, \mathbf{w}_x$ and $\mathbf{Y}^m \, \mathbf{w}_y$ of the variables on these basis vectors is mutually maximized. The correlation coefficient ρ, which is formally defined as

$$\rho = \frac{\mathbf{w}_x^T \mathbf{X} \mathbf{Y}^T \mathbf{w}_y}{\sqrt{\left(\mathbf{w}_x^T \mathbf{X} \mathbf{X}^T \mathbf{w}_x\right)\left(\mathbf{w}_y^T \mathbf{Y} \mathbf{Y}^T \mathbf{w}_y\right)}} \, , \tag{6}$$

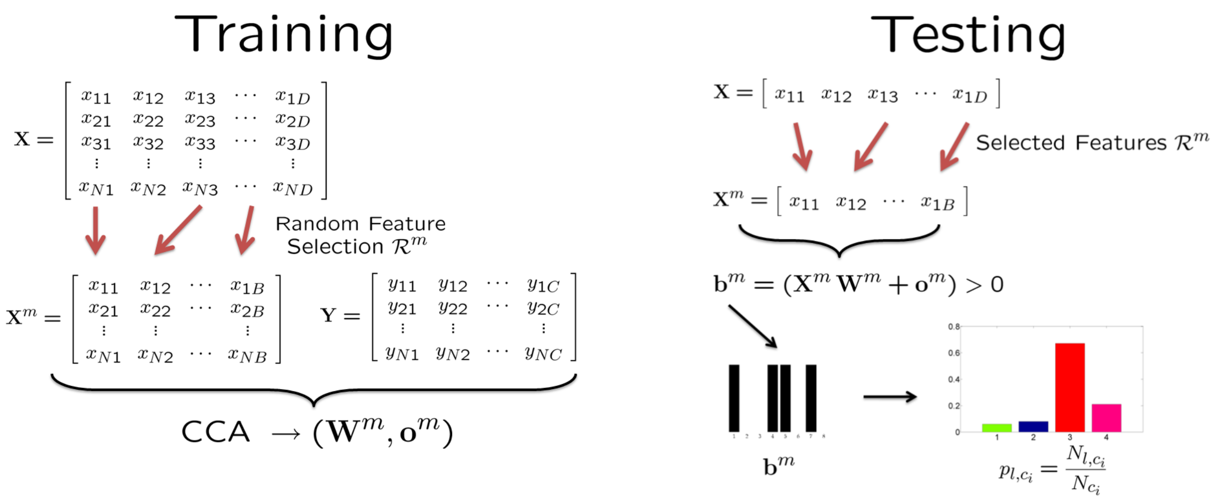

Training
Testing
$\mathbf{X} =$
Random Feature Selection $\mathcal{R}^m$
$\mathbf{X}^m =$
$\mathbf{Y} =$
CCA $\rightarrow (\mathbf{W}^m, \mathbf{o}^m)$
Selected Features $\mathcal{R}^m$
$\mathbf{b}^m = (\mathbf{X}^m\mathbf{W}^m + \mathbf{o}^m) > 0$
$\mathbf{b}^m$
$p_{l,c_i} = \frac{N_{l,c_i}}{N_{c_i}}$

Figure 1: Illustration of our method for a single base classifier. Input data matrix $\mathbf{X}$ is reduced to a fern-specific matrix $\mathbf{X}^m$ by randomly selecting a feature dimension set $\mathcal{G}^m$. This matrix is then used together with the provided label matrix in Canonical Correlation Analysis (CCA) which provides a new embedding space defined by $\mathbf{W}^m$. This projection enables the assignment of each training sample to a bin (using $\mathbf{o}^m$), as well as the calculation of the class-conditional probabilities p_{l,c_i}. During testing the same feature dimensions are selected, and the learned projection is applied to assign the sample to a bin (binary vector $\mathbf{b}^m$). Finally all base classifier probabilities are combined in a semi-naive Bayesian manner.

is maximized. The coefficient ρ is invariant to scaling of the basis vectors and therefore CCA can also be formulated as

$$\begin{aligned} \max_{\mathbf{w}_x,\mathbf{w}_y} \quad & \mathbf{w}_x\mathbf{X}\mathbf{Y}^T\mathbf{w}_y \\ \textit{subject to} \quad & \mathbf{w}_x\mathbf{X}\mathbf{X}^T\mathbf{w}_y = 1\,, \\ & \mathbf{w}_x\mathbf{Y}\mathbf{Y}^T\mathbf{w}_y = 1\,. \end{aligned} \tag{7}$$

To find the basis vectors that maximize the correlation coefficient ρ, one has to solve a generalized eigenvalue problem. We use a regularized version of the eigenvalue problem to prevent overfitting by solving

$$\mathbf{X}\mathbf{Y}^T\left(\mathbf{Y}\mathbf{Y}^T + r_y\mathbf{I}\right)^{-1}\mathbf{Y}\mathbf{X}^T\mathbf{w}_x = \lambda_x\left(\mathbf{X}\mathbf{X}^T + r_x\mathbf{I}\right)\mathbf{w}_x \tag{8}$$

where r_y and r_x are the two regularization parameters and I is the identity matrix. As recently shown in [22], regularization of $\mathbf{Y}$ by r_y does not affect the projection of $\mathbf{X}$, therefore we set $r_y = 0$. According to [22], we fixed r_x to 10^{-6} in all our experiments.

We consider the S leading eigenvectors obtained from the solution of the generalized eigenvalue problem defined in Equation 8 as our $B \times S$ projection matrix $\mathbf{W}^m$, which allows to project any new data point into this novel embedding space. We define the split values independently at each dimension, by simply thresholding the projected values at the median of the training data values, yielding the offset vector $\mathbf{o}_m$. In this way, we finally obtain the required $S \times 1$ binary vector $\mathbf{b}^m$ by applying Equation 5, which uniquely assigns data points to one of the 2^S bins. Figure 1 illustrates our embedded fern concept for a single fern.

By repeating the previously described steps we obtain a set of random ferns defined by their projection matrices and the offset vectors $(\mathbf{W}^m, \mathbf{o}^m)$. Please note that for obtaining our projection matrices, the reduced feature matrix $\mathbf{X}^m$ differs for each fern, while the label matrix $\mathbf{Y}$ is the same for all CCA steps. Hence, in contrast to conventional random ferns, we only inject randomness by selecting the feature dimensions for each fern, since the mapping is then estimated in a deterministic manner per fern.

The final mappings to the binary vectors $\mathbf{b}^m$ are then used to define the class conditional probabilities p_{l,c_i} by counting the number of class members that evaluate to the corresponding bins. During testing the data points are mapped to the different projection spaces per fern using $(\mathbf{W}^m, \mathbf{o}^m)$ in Equation 5, which provides the class conditional probabilities and the final classification results by combining the fern probabilities in a semi-naive Bayesian manner as described in Equation 3.

Comparison to Fern and Forest Obviously our proposed approach shares many similarities with random forests and ferns. All three methods follow the same principle: (a) using randomness to separate the feature space into a number of $L = 2^S$ bins per forest (fern), (b) counting the number of labeled training data examples reaching the different bins in order to define class conditional probabilities and (c) building an ensemble of these classifiers in order to make the final decision.

Since all of its tests are selected randomly, the random fern has the lowest computational complexity of $O\left(N\,log\,L\right)$ for training. The training for random forests is much slower by comparison, since in order to obtain reasonable results many node test candidates have to be evaluated by for example an information gain measure, which leads to a complexity of $O\left(K\,N\,L\right)$. The training of CCA ferns is slower than that of random ferns, since we have to additionally solve the generalized eigenvalue problem for CCA. However, only a single CCA calculation is required for each fern and no information gain tests have to be done, which makes the training of CCA ferns much faster than that of random forests. The testing is efficient for all three methods, having a complexity of $O\left(log\,L\right)$ for conventional forests and ferns and a slightly higher complexity of $O\left(B\,log\,L\right)$ for our CCA ferns, since we have to additionally project the data points. However, since B can be chosen much smaller than the feature dimensionality (see experiments), our method requires only slightly more computation time during testing.

Considering the memory profile, we have to save for each fern the $B \times S$ projection matrix $\mathbf{W}^m$ and the $S \times 1$ vector $\mathbf{b}^m$. In addition, we have to save $F \times L \times C$ log probability values. The maximum number of non-zero values in the probability table equals $F \times N$ (if each sample falls in a different bin). However, in practice the probability table is much sparser (as also shown in the experiments). Hence, in general we have a smaller memory profile than standard brute-force matching methods, that have to keep all training samples for testing.

5. Experiments

Experiments demonstrate the principle usefulness of discriminative classification for localization and the improved performance of our proposed embedded ferns. We first evaluate in Section 5.1 the obtainable classification performance in the scenario of image-based localization on two widely used datasets (*Dubrovnik* and *Rome*) comparing our results to a conventional nearest neighbor approach and to related randomization methods. In Section 5.2 we analyze the influence of the main parameters on performance.

5.1. Comparison to Related Methods

We formulate a classification task in the scope of image-based localization for our experimental evaluation. We assume that we have given a 3D reconstruction of the area we are interested in, obtained by any structure-from-motion pipeline. Hence, we have given all camera poses, the 3D coordinates of the sparse point cloud, a list of all image points that were used for triangulating the 3D points and all corresponding descriptors.

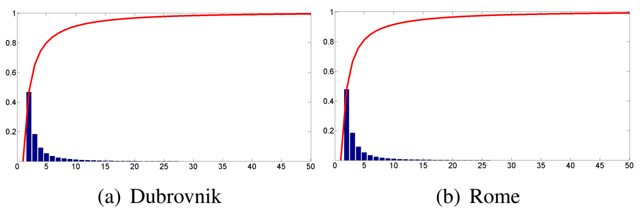


Figure 2: Statistics for image-based localization datasets. Bar plot shows the percentage of overall 3D points (y-axis) that have the corresponding number of 2D points (x-axis). Red line shows the cumulative sum over the percentage.

We use the two most popular datasets in the field of image-based localization *Dubrovnik* and *Rome* [14] for evaluation. Both datasets consist of photos taken from Flickr and the sparse 3D point models were obtained by Bundler [21]. *Dubrovnik* consists of approximately 6K images, 2M points and 10M descriptors, whereas *Rome* approximately consists of 15K images, 4M points and 21M descriptors. In both datasets we identify potentially visible sets [1] from the sparse 3D point clouds, which divide them into several smaller view cells. The potentially visible sets determine the subset of 3D points and the corresponding descriptors that have to be considered within the current view cell. We do this according to our intended scenario, where we approximately know the current location using for example GPS.

Since we exploit all available 2D descriptors per 3D point for discriminative feature-to-point matching, the overall distribution of the number of descriptors per point is of interest. Figure 2 shows histograms for the two datasets considered. As can be seen, the distributions are quite similar. On average we have 5 descriptors per 3D point and 20% of the 3D points have more than five 2D representations. Hence, for many 3D points large sets of descriptors are available, and our classifier tries to use all available information in a discriminative manner. Especially in an image-based localization task, the 3D points with the largest number of corresponding 2D descriptors are the landmark points (like a tower) that are most important for localization. As a consequence, improving matching quality for these points is especially important.

We first demonstrate that analyzing all available 2D descriptors in a discriminative manner allows to improve matching performance. We evaluate the performance when using our proposed classifier to identify the feature-to-point matches in comparison to a standard nearest neighbor matching approach. We stick to a K-fold cross validation

	Dubrovnik	*Rome*
Nearest Neighbor	$96.06 \pm 1.22\%$	$95.53 \pm 1.10\%$
Random Fern	$92.65 \pm 1.73\%$	$88.12 \pm 1.72\%$
Random Forest	$96.31 \pm 1.57\%$	$89.18 \pm 1.31\%$
Proposed	$\mathbf{98.97 \pm 0.89\%}$	$\mathbf{97.64 \pm 0.76\%}$

Table 1: Comparison of obtainable matching performance using different approaches. Mean classification accuracies and standard deviations for 5-fold cross validation experiment per potentially visible set are shown.

(with $K = 5$) for each visible set. This means that we train our classifiers on the available data from $K - 1$ folds, where we removed all 2D descriptors seen in the remaining fold. The ground truth labels correspond to the IDs of the 3D points. During testing we then try to find the correct feature-to-point correspondences using all features of the remaining fold. In such a way, we do have ground truth labels for each query image feature, i. e. we are aware of the corresponding 3D point it should classify to. This process is repeated for all K folds and we evaluate the average classification accuracy, which correlates to the expected feature-to-point matching performance.

We compare our approach to a conventional approximated nearest neighbor search and to the related randomization methods random ferns and random forests using Matlab implementations. The nearest neighbors are identified using KD-trees. For the randomization methods we use publicly available reference implementations from the toolbox of Piotr Dollar [6]. We use the same number of base classifiers $F = 50$ for all methods. The random fern uses 10 binary decisions per base classifier. The trees in the random forest are all fully grown and we fix the number of evaluated node tests to $\sqrt{D}$ [8]. For our proposed classifier we fix S to 16 and B to 25. We also implemented our method in Matlab and code is available at http://vh.icg.tugraz.at.

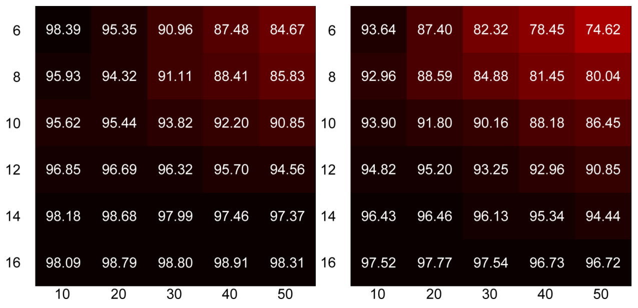
Table 1 shows the average classification accuracies and standard deviations on both datasets. As can be seen by additionally considering the available information in a discriminative setting, we are able to outperform a standard nearest-neighbor approach and conventional random ensemble approaches in this scenario. We additionally analyze the runtime and the memory profile required for identifying the correspondences. The runtimes for the classifiers are constant and independent of the number of 3D points, where our approach requires on average 1.04 times longer than the fern (due to the mapping) and the random forest is slightly slower (1.30 times). The runtime of the nearest neighbor approach depends on number of 3D points, starting to being slower if we have a more than a few hundred 3D points in our database. Considering memory profile, our approach requires $F \times C \times 2^S$ bytes for saving the probability tables. However, the tables are extremely sparse, where on average only 1.17% of the bins are non-zero leading to a constant and reasonable memory profile.

Query images not used for the 3D reconstruction can now be localized by evaluating the obtained correspondences using a perspective-n-point (PnP) algorithm in a hypothesize-and-verify framework such as PROSAC [4]. In such a way we are able to estimate an accurate camera pose for each query image. Unfortunately, the increased matching performance does not directly map to an improved localization quality on these datasets, however our approach has implicitly several advantages to a standard BF approach like (a) consistent runtime independent of point cloud size, (b) small memory profile since we do not have to save the training data, (c) implicit probabilistic ranking of correspondences, (d) anytime capability, since we can stop after evaluating only a subset of the base classifiers and (e) improved matching performance as shown in the experiments due to exploiting all available information.

5.2. Parameter Evaluation

As second experiment we aim at analyzing the influence of parameters on performance. We have three different parameters to fix: the number of base classifiers (ferns) F, the number of features per fern B and the number of bins per fern $L = 2^S$. We use the same datasets as aforementioned but only select a subset (5 per dataset) of the potentially visible sets for evaluation due to runtime issues. It is well known that increasing the number of base classifiers F improves recognition performance until some kind of saturation is achieved. Similar behavior can be observed from our method where the saturation point is approximately achieved when using 50 base classifiers.

We further analyze dependencies between the depth S and the number of feature dimensions B that are used to find the Canonical Correlation Analysis (CCA) projection. Figure 3 illustrates the influences on our datasets, adapting the depth of our embedded fern from 6 to 16, and the number of considered feature dimensions from 10 to 50. As expected, to maintain the randomization principle is important to select only a subset of feature dimensions for each base classifier (~ 20). Similar as for standard random forests, the number of binary splits has to be adapted to the amount of training samples to avoid overfitting. Since in our scenario a large number of training samples is available, larger numbers lead to the best results.



	10	20	30	40	50
6	98.39	95.35	90.96	87.48	84.67
8	95.93	94.32	91.11	88.41	85.83
10	95.62	95.44	93.82	92.20	90.85
12	96.85	96.69	96.32	95.70	94.56
14	98.18	98.68	97.99	97.46	97.37
16	98.09	98.79	98.80	98.91	98.31

(a) Dubrovnik

	10	20	30	40	50
6	93.64	87.40	82.32	78.45	74.62
8	92.96	88.59	84.88	81.45	80.04
10	93.90	91.80	90.16	88.18	86.45
12	94.82	95.20	93.25	92.96	90.85
14	96.43	96.46	96.13	95.34	94.44
16	97.52	97.77	97.54	96.73	96.72

(b) Rome

Figure 3: Parameter influence on classification results for two different datasets. On y-axis: the number of binary splits (from 6 to 16). On x-axis: the number of randomly selected feature dimensions (from 10 to 50).

6. Conclusion

In this paper we analyzed the applicability of discriminative classifiers for image-based localization. We demonstrated that such a formulation has several advantages to a standard nearest neighbor assignment approach like a runtime independent of the point cloud size, a constant memory profile, where we do not have to save the point cloud descriptors, and an improved matching performance by using all available information in a discriminative manner. Because of the specific demands in such a localization scenario, standard classifiers are not well suited. For this reason, we additionally proposed an extension of the classical random fern principle, adding a discriminative embedding step per base classifier which directly yields flat assignments to bins. Experiments show that our method yields higher classification rates in comparison to random ferns and forests in an image-based localization scenario.

Acknowledgment

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 601139 CultAR (Culturally Enhanced Augmented Realities).

References

[1] C. Arth, D. Wagner, M. Klopschitz, A. Irschara, and D. Schmalstieg. Wide area localization on mobile phones. In *ISMAR*, 2009.

[2] S. Cao and N. Snavely. Graph-based discriminative learning for location recognition. In *CVPR*, 2013.

[3] S. Choudhary and P. J. Narayanan. Visibility probability structure from sfm datasets and applications. In *ECCV*, 2012.

[4] O. Chum and J. Matas. Matching with PROSAC - progressive sample consensus. In *CVPR*, 2005.

[5] A. Criminisi, J. Shotton, and E. Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends in Computer Graphics and Vision*, 7:81–27, 2012.

[6] P. Dollár. Piotr's Image and Video Matlab Toolbox (PMT). `http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html`.

[7] J. Gall and V. Lempitsky. Class-specific Hough forests for object detection. In *CVPR*, 2009.

[8] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *ML*, 63:3–42, 2006.

[9] P. Gronat, G. Obozinski, J. Sivic, and T. Pajdla. Learning and calibrating per-location classifiers for visual place recognition. In *CVPR*, June 2013.

[10] D. Heath, S. Kasif, and S. Salzberg. Induction of oblique decision trees. *Journal of Artificial Intelligence Research*, 32, 1993.

[11] H. Hotelling. Relations between two sets of variates. *Biometrika*, 28:312–377, 1936.

[12] A. Irschara, C. Zach, J. M. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *CVPR*, pages 2599–2606, 2009.

[13] C. Leistner, M. Godec, S. Schulter, A. Saffari, M. Werlberger, and H. Bischof. Improving classifiers with unlabeled weakly-related videos. In *CVPR*, 2011.

[14] Y. Li, N. Snavely, and D. Huttenlocher. Location recognition using prioritized feature matching. In *ECCV*, 2010.

[15] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3D point clouds. In *ECCV*, 2012.

[16] H. Lim, S. N. Sinha, M. F. Cohen, and M. Uyttendaele. Real-time image-based 6-DoF localization in large-scale environments. In *CVPR*, 2012.

[17] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006.

[18] M. Özuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *PAMI*, 32, 2010.

[19] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2D-to-3D matching. In *ICCV*, 2011.

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

[21] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. *ACM Transactions on Graphics*, 25(3):835–846, July 2006.

[22] L. Sun, S. Ji, and J. Ye. Canonical correlation analysis for multilabel classification: A least-squares formulation, extensions, and analysis. *PAMI*, 33:194–200, 2011.

[23] B. Yao, A. Khosla, and L. Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *CVPR*, 2011.

[24] W. Zhang, X. Wang, and X. Tang. Coupled information-theoretic encoding for face photo-sketch recognition. In *CVPR*, 2011.