

Adding Cues to Binary Feature Descriptors for Visual Place Recognition

Dominik Schlegel and Giorgio Grisetti

Abstract—In this paper we propose an approach to embed continuous and selector cues in binary feature descriptors used for visual place recognition. The embedding is achieved by extending each feature descriptor with a binary string that encodes a cue and supports the Hamming distance metric. Augmenting the descriptors in such a way has the advantage of being transparent to the procedure used to compare them. We present two concrete applications of our methodology, demonstrating the two considered types of cues. In addition to that, we conducted on these applications a broad quantitative and comparative evaluation covering five benchmark datasets and several state-of-the-art image retrieval approaches in combination with various binary descriptor types.

I. INTRODUCTION

Visual Place Recognition (VPR) has been receiving increased attention over the last decade [1]–[4]. The task of a VPR system is the retrieval (i.e. recognition) of similar images from a database of visited places that match a current image. This *similarity search* problem is typically addressed through image comparison. To reduce the dimensionality of the problem and to gain robustness to viewpoint and illumination variations, it is common to represent an image by a set of *feature descriptors* [5]–[15]. These descriptors are generally represented by floating-point or binary *vectors* that admit a certain distance metric. Comparing two images is then reduced to measuring the cumulative distance between their corresponding descriptors.

Typically, descriptors are computed exclusively from image pixel data. On one hand, this allows to use these descriptors on any system that needs VPR capabilities. On the other hand, if external place-specific *cues* (e.g. GPS coordinates) are available, the VPR system based on descriptors cannot immediately benefit from that additional information. The cues can be incorporated in a pre- or post-filtering stage. In general however, this strategy requires a major modification of the similarity search approach [16]–[18].

Alternatively, one can *embed* the cues as additional dimensions in the descriptor vector, without modifying the search approach. During the search, the additional cues will contribute to the distance metric, resulting in small cumulative distances when the described image portions *and* the cues are similar [19]–[23].

Embedding continuous cues is straightforward when the descriptors are floating-point vectors. Since in this case the common distance metric is the L_2 -norm. Conversely, *binary descriptors* are compared with the Hamming distance L_H [24], that is the number of mismatching bits between

Both authors are with the Department of Computer, Control and Management Engineering, Sapienza University of Rome, Rome, Italy
`{lastname}@diag.uniroma1.it`

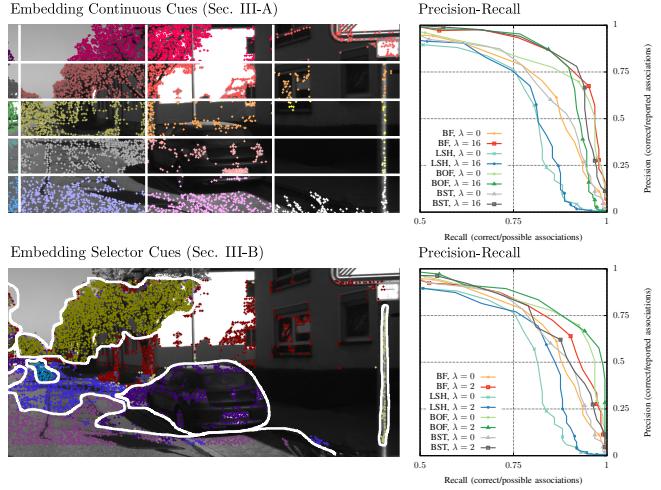


Fig. 1: Image extracts with FAST keypoints and BRIEF-256 descriptors, accompanied by a Precision-Recall (PR) evaluation. Top: Keypoint coordinates as additional cues. Bottom: Semantic labels as additional cues.

the compared descriptors. A *binary cue* can be added to a binary descriptor, analogously to the floating-point case. A continuous cue, however, cannot be added straightforwardly to a binary descriptor since the Hamming distance between two floating-point values does not reflect their arithmetic difference, unless they are equal. Thus the binary representation of a floating-point value does not qualify as a valid binary feature for the cue that can be used to augment the descriptor.

In this paper, we address the problem of adding *continuous* and *selector cues* to binary descriptors. We propose an approach to compute binary features from continuous cues which can be used to augment the descriptors. These binary strings support the Hamming distance, thus comparison approaches that are based on this norm do not have to be changed to deal with the augmented descriptors. Additionally, we elaborate on how to encode a selector cue. Selector cues assume a specific value from a finite set and the distance between two such cues is either 0 if they are the same or 1 otherwise. A typical type of selector is the semantic class of a pixel returned by a classifier.

We demonstrate the applicability of our approach for two example cues and verify its benefit for VPR in an extensive experimental evaluation on five public benchmark datasets with several state-of-the-art VPR approaches. In Fig. 1 we display a comparative performance evaluation of our example cues on the KITTI Simultaneous Localization and Mapping (SLAM) dataset. All results presented in this paper can be reproduced using our open-source C++ implementation¹.

¹Source at: www.gitlab.com/srrg-software/srrg_bench

II. RELATED WORK

Many efforts have been made to improve existing, hand-crafted local binary descriptors, creating the broad variety available today [25]. Years before the introduction of the well-known BRIEF binary descriptor [8], Mikolajczyk and Matas [19] presented an approach to improve the state-of-the-art SIFT [6] floating point descriptor for matching. In their work, they transformed the original descriptors based on learned optimal projections under the Mahalanobis distance. Another early work about enhancing scene classification and object recognition for various global and local descriptors has been reported by Harada *et al.* [20]. Harada *et al.* showed that by consideration of locality, correlation and the concatenation of local descriptors the matching accuracy can be raised significantly. None of these works considered the local binary descriptors which we evaluate in this paper.

Shortly after the release of the pioneering BRIEF binary descriptor, the well-known ORB [9] and BRISK [10] descriptors were introduced. ORB improved BRIEF by adding rotation and scale invariance, while BRISK improved BRIEF by considering a Gaussian pixel average for the descriptor computation. After the introduction of the bio-inspired FREAK binary descriptor [12], Kottman [21] presented with LFREAK an enhanced version of FREAK by including the spatial arrangement between multiple keypoints. Similarly, Wang *et al.* [22] proposed with CS-FREAK a FREAK variant that considers the neighborhood intensity information of its surrounding sampling points. Recently, Xiao *et al.* [23] introduced a binary descriptor (BAG) suitable for RGB-D image processing. Using binary tests, Xiao *et al.* add geometric cues based on point depth to the local binary pattern based on image intensity in a patch. In contrast to our work, their cues are not computed additively to the descriptor and neither transfer quantifiable cue distances.

For the remainder of this article, if not specified otherwise, with *descriptors* we always refer to local binary descriptors. Binary descriptors are generally compared by their Hamming distance. Sankaran *et al.* [18] investigated the effects of a weighted Hamming distance and a thresholded binary testing on ORB. Sankaran *et al.* show that an improved accuracy can be achieved on various datasets.

Often, these descriptor improvements are tailored to a specific search approach for a specific descriptor type. Among the most popular similarity search approaches is the Bag-of-Features (BOF) approach of Sivic and Zisserman [26]. BOF reduces the high dimensionality of the search problem by quantization of descriptors into *visual words*, for which a frequency histogram can be obtained that describes an entire image. Subsequently, Jegou *et al.* [17] presented a twofold improvement of the BOF approach. By differentiation on the position of a local descriptor within its k-means cluster, and adding a consistency check on the keypoint's angle and scale, precision and runtime of BOF could be improved significantly. The euclidean descriptor positions are encoded into binary signatures, with a procedure introduced as Hamming Embedding. In contrast to their procedure,

which converts descriptors into binary strings, our approach generates binary strings based on auxiliary information from an arbitrary source, that are added to a binary descriptor. Furthermore, our approach does not require a learning phase and is completely independent of the target similarity search approach. In their Bags of Binary Words (DBoW) open-source library, Gálvez-López and Tardós [2] implemented the BOF approach and added further improvements while making it accessible for the SLAM research community.

Multi-probe Locality Sensitive Hashing (LSH) by Lv *et al.* [16] is a popular similarity search approach, that reduces the dimensionality of the search problem with hashing. Depending on the chosen hash key lengths and the number of hashing tables, LSH requires significantly more memory than BOF to index descriptors in its database.

In Hamming Binary Search Tree (HBST), one of our works [27], we presented a Binary Search Tree (BST) approach for fast binary descriptor search. By arranging the descriptors in a tree based on particular bits, the search problem dimensionality is reduced by one and the number of search candidates is halved, for each traversed node. HBST achieves excelling search speed, while maintaining a meaningful accuracy in small and large scale scenarios.

Note that the mentioned similarity search approaches [16], [26], [27] are only guaranteed to find an *approximate nearest neighbor*, as opposed to the exhaustive Brute-Force (BF) search, which always returns the best match.

The aim of this paper is to present an approach that improves the VPR image retrieval and descriptor matching precision of an existing similarity search system by considering additional, place-relevant cues. While most of the discussed works require a modification of the descriptor computation or similarity search method, our approach is purely supplementary. Additionally, our approach has a negligible memory footprint and comes at a vanishing computational cost. Since we exploit the particular conditions of VPR to our advantage, our approach is solely targeted at VPR.

III. OUR APPROACH

In this paper we address image retrieval based on feature descriptor matching. More specifically, to obtain a measure for the similarity between images, we compare their corresponding descriptors.

Feature descriptors are vectors that encode the local appearance of an image around a point of interest (keypoint). Floating-point descriptors are vectors of continuous numbers and are usually compared with the L_2 -norm. Binary descriptors are stored in binary vectors and are compared using the Hamming distance L_H [24]. Descriptors are computed so that the distance between them grows with the dissimilarity between the corresponding, described image regions.

Binary descriptors generally occupy significantly less memory (128 to 512 bits) than their floating-point counterparts (512 to 4096 bits) and are often cheaper to compute. Furthermore, binary descriptors can be compared much faster than floating-point descriptors, since the Hamming distance can be efficiently computed on modern CPUs. Additionally,

state-of-the-art binary descriptors such as [8]–[12] are more accessible than state-of-the-art floating point descriptors, which are subject to patents pending [6]. For these reasons binary descriptors are generally preferred to floating-point descriptors for **VPR** or **SLAM** applications [28]–[30].

Binary descriptors are computed based on local intensity properties of the image, and are targeted at *image recognition*. They do not encode information originating from additional cues, that are not necessarily present in the image. In **VPR** applications one often obtains such cues (e.g. point depth), that can be used to verify recognized image candidates in a postprocessing phase. In the following, we define an approach for adding such continuous or integer cues to binary descriptors, to capture this additional information directly in the descriptor. The benefit of extending existing descriptors is that the remaining part of the system originally thought to deal with binary descriptors does not have to be changed to work with the additional cues, except for the length of the descriptors considered.

A. Converting Continuous Cues into Binary Strings

Without loss of generality, let c be a continuous value in the interval $[0, 1)$ encoding a cue that we want to add to a binary descriptor \mathbf{d} . If c is not contained in the target range, one can use an affine operator $\bar{c} = \alpha c + \beta$ such that the values of \bar{c} lie in the specified range. For the sake of notation, in the remainder of this section we assume c to be normalized for the target range.

We aim at converting the value c into a binary string $\mathbf{b} = b(c)$, that appended to the original descriptor \mathbf{d} , results in a new descriptor $\mathbf{d}_* = \langle \mathbf{d}, \mathbf{b} \rangle$. Equally to \mathbf{d} , also \mathbf{d}_* is compared with the Hamming distance. Thus, \mathbf{b} must be Hamming distance compatible as well. Furthermore, to transfer the gathered discrimination between two values c and c' , we need to ensure that the Hamming distance between the binary strings \mathbf{b} and \mathbf{b}' computed from c and c' , grows monotonically with the distance between c and c' . These requirements can be expressed by the following equation:

$$L_H(b(c), b(c')) \propto |c - c'| . \quad (1)$$

To compute $b(c)$ we quantize the range $[0, 1)$ in I even intervals, each of length $1/I$. For such a quantization, we can retrieve a binary string $\mathbf{b} = b(c)$ consisting of $I - 1$ bits, according to:

$$b(c) = \langle b_0(c), b_1(c), \dots, b_{I-2}(c) \rangle . \quad (2)$$

Where the value of the bit $b_i(c)$ is 1 if c lies in an interval higher than the interval at $i + 1$, 0 otherwise. More formally:

$$b_i(c) = \begin{cases} 1 & \text{iff } c > \frac{1}{I}(i+1) \\ 0 & \text{otherwise} \end{cases} \quad i \in \{0, 1, \dots, I-2\} . \quad (3)$$

Fig. 2 illustrates the proposed procedure with a quantization of $I = 5$ intervals and two example cue conversions.

The above procedure can be applied also when the domain of the cues is multi-dimensional. In that case, each cue c_n will contribute with an independent binary string \mathbf{b}_n to \mathbf{d}_* .

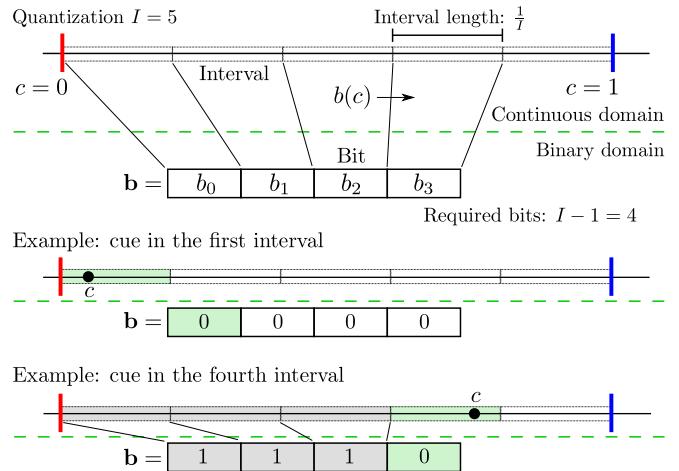


Fig. 2: Top: Visualization of the proposed quantization and assignment function described in Eq. (2) with a quantization in $I = 5$ intervals. Center and bottom: Example conversion for two different cue values. Note that we do not have to set any bit for cues in the first interval (center).

The Hamming distance between the augmented descriptors $\mathbf{d}_*, \mathbf{d}'_*$ of multi-dimensional cues is proportional to the Manhattan distance in the continuous space. In the following, we provide a straightforward example for transforming two-dimensional cues according to our procedure.

Example - Converting Keypoint Coordinates (KC): In relevant **VPR** applications, such as autonomous cars, images of the same place are often acquired from viewpoints similar to each other. Feature detectors, such as the popular FAST corner detector [31], are constructed to return similar keypoint detections for similar images. Hence, we analyze the effect of adding the keypoint coordinates (u, v) to the descriptors w.r.t. the achieved **VPR** accuracy.

To convert this two-dimensional cue, we need only to define the vertical quantization I_v and horizontal quantization I_u of the keypoint coordinates (u, v) . In Fig. 3 we display a two-dimensional image quantization of $I_u = 5, I_v = 3$ and the resulting binary string $\mathbf{b} = b(c)$ computed with Eq. (2), for an example keypoint.

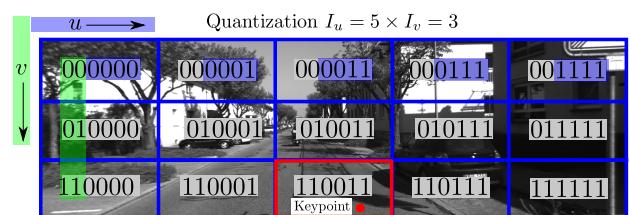


Fig. 3: Proposed two-dimensional cue conversion based on a keypoint position $c = (u, v)$ (red dot) in a $I_u = 3 \times I_v = 5$ quantization scenario. The obtained binary strings are $\mathbf{b}_v = b(c_v) = \langle 1, 1 \rangle$ and $\mathbf{b}_u = b(c_u) = \langle 0, 0, 1, 1 \rangle$, resulting in the composite string $\mathbf{b} = \langle 1, 1, 0, 0, 1, 1 \rangle$ (red box).

Depending on the chosen quantization, higher distances will result in the dimension with the most partitions. This enables us to stress horizontal position similarities over vertical ones. Notably, the quantization and the binary string mapping can be computed once at startup for all possible cues (i.e. pixels) on the image plane. This enables us to transform a keypoint position with a *single lookup* at runtime.

We validate the effectiveness of our additional cue for **VPR** in a series of experiments (Sec. IV).

B. Converting Selector Cues into Binary Strings

In the previous section we focused on embedding *continuous values* into binary descriptors. Here we describe a procedure to embed an arbitrary *selector value*.

A selector value i can be mapped to the finite integer range $\mathcal{I} = \{0, 1, \dots, I - 1\}$ and typically encodes *discrete cues* such as label information. Let the distance between two selectors be either 2 if they are different, or 0 if they are identical. This distance property can be straightforwardly implemented with the Hamming distance by using a binary string \mathbf{b} of length I bits. A cue c of value $i \in \mathcal{I}$ is represented by a binary string $\mathbf{b} = \langle b_0(c), b_1(c), \dots, b_{I-1}(c) \rangle$, where the bit $b_i(c)$ is set to 1, and all others to 0.

Example - Converting Semantic Labels (SL): In the remainder of this section we provide an example on how to use selector cues within a **VPR** system based on binary descriptors. To this extent we add to each descriptor the semantic label computed at its keypoint position. We retrieve the label by using the SegNet [32] system². We ran SegNet with an off-the-shelf configuration (webdemo) for which we can recognize up to 12 selector cues. Fig. 4 illustrates the described selector mapping for an example image.

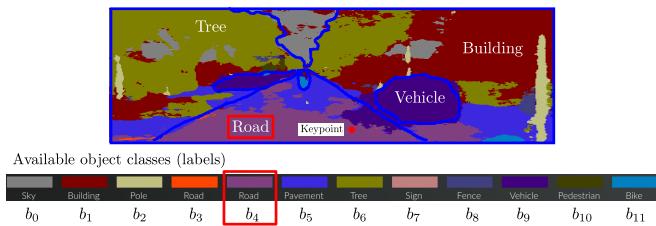


Fig. 4: Proposed cue conversion based on the label (1 of 12) at the keypoint location (red dot). In this scenario, the example keypoint is labeled as: Road, resulting in the string $\mathbf{b} = \langle 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0 \rangle$ (red box).

Albeit the segmentation was not always correct, the precision of the **VPR** system improved, as reported in our experiments (Sec. IV). We remark that for **VPR**, we do not necessarily rely on *correct* labels as long as the labeling is *consistent* (e.g. if the network mistakenly labels a vehicle as a tree in a place, and as we return to that place, does the same mistake again, we still have consistent similarity search information). Consistency is easier to achieve than correctness, since the latter implies the former.

In contrast to the continuous cues of the **Keypoint Coordinates (KC)** of the previous example, the **SL** labels are *rotation and translation insensitive*. Hence, images acquired from different viewpoints can still be matched.

C. Augmentation Weighting

When the Hamming distance $L_H(\mathbf{d}, \mathbf{d}')$ between two descriptors \mathbf{d}, \mathbf{d}' is smaller than a certain threshold $\tau \geq 0$, \mathbf{d}, \mathbf{d}' are assumed to originate from the same image feature (*match*). Typical binary descriptors have bit sizes

of 256 or 512. Depending on the desired continuous range quantization, the resulting auxiliary binary string \mathbf{b} can be much smaller (e.g. 6 bits) than the descriptor \mathbf{d} . Hence, the augmented descriptor $\mathbf{d}_* = \langle \mathbf{d}, \mathbf{b} \rangle$ might change only very little in size. In such a case, the cues' contributions to \mathbf{d}_* respectively \mathbf{d}'_* vanish when the Hamming distance is computed, more formally: $L_H(\mathbf{d}_*, \mathbf{d}'_*) \approx L_H(\mathbf{d}, \mathbf{d}')$.

To tackle this issue, we introduce a variable *augmentation weight* $\lambda \geq 0$ and define the *weighted Hamming distance* as:

$$L_H^\lambda(\mathbf{d}_*, \mathbf{d}'_*) = L_H(\mathbf{d}, \mathbf{d}') + \lambda L_H(\mathbf{b}, \mathbf{b}') \quad (4)$$

It is easy to see that for $\lambda \rightarrow 0$ the contribution of the added cue \mathbf{b} vanishes, whereas for $\lambda \rightarrow \infty$ the original descriptor \mathbf{d} is not considered anymore. Instead of modifying the search method to compute the weighted Hamming distance, Eq. (4) can also be achieved by appending an augmentation λ times to a descriptor (assuming λ is an integer). In our experiments we conducted such an evaluation, without touching the comparison function of the search methods. We list the values of λ which we found working best for our two example cues **KC** and **SL**.

IV. EXPERIMENTAL EVALUATION

The focus of this work is to improve the precision of feature-based similarity search methods for **VPR** by adding cues (Sec. III-A, Sec. III-B) to descriptors without requiring major adjustments for the search methods. In the following we introduce the descriptors (Sec. IV-A), the search methods (Sec. IV-B), the performance metrics (Sec. IV-C) and the various datasets (Sec. IV-D) considered in our experiments. Subsequently, we display and discuss our results (Sec. IV-E).

A. Descriptors

Our cue embedding strategies (Sec. III-A, Sec. III-B) and their beneficial effects for a similarity search method are independent of the selected binary descriptor. To support this claim, we conducted experiments on several of the most common local binary descriptors:

BRIEF [8] was one of the first local binary descriptors to enable efficient computation, storage and comparison. Due to its viability for real-time applications, BRIEF is broadly used to this day.

ORB [9] improved BRIEF by adding rotational and scale invariance. It is widely used in feature-based Visual Odometry (VO) systems.

BRISK [10] improved BRIEF by considering a Gaussian pixel average for the descriptor computation.

A-KAZE [11] is an accelerated version of the KAZE descriptor. KAZE captures image regions in a nonlinear scale space, which comes at a high computational cost. By using a more recent scheme to build the nonlinear scale space, the significant acceleration was obtained.

FREAK [12] is one of few bio-inspired descriptors. Its binary signature is computed by efficiently comparing image intensities over a retinal sampling pattern.

LDAHash [13] compresses SIFT [6] descriptors in binary descriptors through a supervised binarization scheme.

²SegNet: www.github.com/alexgkendall/caffe-segnet

BinBoost [14] is an extremely compact binary descriptor (as small as 8 bits) where each individual bit is computed by a learned hash function. The hash functions are learned such that each bit complements the others.

We utilized the current OpenCV³ (3.4.7) implementation of all descriptor types except for LDAHash where we integrated the released C++ source code by the authors.

B. Similarity Search Methods

We tested our augmented descriptors on 4 state-of-the art feature-based similarity search methods (Sec. II). Each one of them having its advantages and drawbacks.

BF: The straightforward, exhaustive search approach, guaranteeing the best matches at the highest computational cost. We utilized the official implementation of the publicly available OpenCV library (3.4.7).

LSH: Fast Library for Approximate Nearest Neighbors with multi-probe Locality-sensitive hashing. We utilized the official implementation of the OpenCV library (3.4.7).

BOF: State-of-the-art method for fast place recognition, employed by many current **SLAM** systems. A well maintained and widely used **BOF** library in conjunction with binary descriptors is **DBoW** [2]. We utilized the authors' publicly available implementation⁴.

BST: A binary search tree approach, suitable for highly efficient and lightweight similarity search in large scale datasets with binary descriptors. We utilized the publicly available **HBST** [27] implementation⁵.

C. Performance Metrics

We chose to examine multiple of the most common performance metrics used in **VPR**:

Precision-Recall (PR): To determine the reliability of a **VPR** system one generally measures the resulting Precision and Recall statistics. The first being:

$$Precision = \frac{\# \text{ correctly reported matches}}{\# \text{ total reported matches}} \in [0, 1].$$

Regarding the completeness of the results, one considers:

$$Recall = \frac{\# \text{ correctly reported matches}}{\# \text{ total possible correct matches}} \in [0, 1].$$

Here, *match* refers to a pair of images that are reported by the **VPR** system to originate from the same place.

Mean Average Precision (mAP): The **mAP** reflects the average Precision of *multiple PR* curves over equidistant Recall levels. With the **mAP** one can concisely describe the achieved performance of an approach over a *series of test cases* in a single number.

Mean image processing time \bar{t} : To examine the impact of augmented descriptors on the computational cost of a search method, we measure the required time for matching a set of descriptors of an image and integrating it into the database.

For the Oxford and the Paris datasets (Sec. IV-D), we concisely present our results using **mAP** (for which the

authors specifically provide a tool [33]). We adapted the tool to also compute the **mAP** for the ZuBuD and the Holidays dataset. For KITTI we present individual **PR** curves.

D. Datasets

For evaluating feature-based **VPR** similarity search methods, one generally considers a large number of reference images, of which only few describe the same place as a query image. Many of the images serve purely to perturb the search method (*distractor images*). Ground truth information is generally provided by listing the matching query to reference image pairs (*correctly reported matches / true positives*). This data is needed to compute the achieved **PR** and **mAP** performance indicators (Sec. IV-C).

Fig. 5 displays an image sample for each of the five datasets we considered. All of them are publicly approved and well-used standard datasets for evaluating the performance of a **VPR** system.

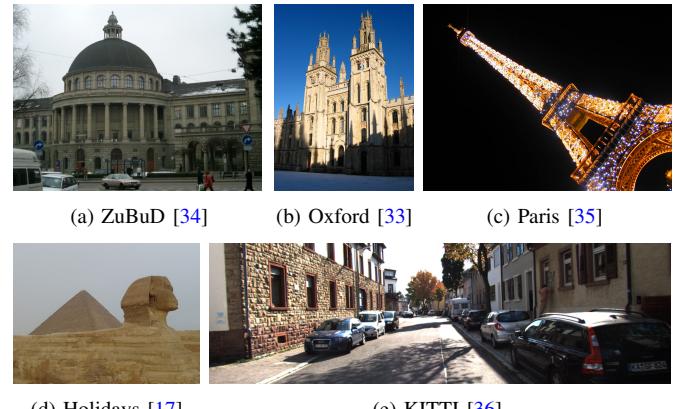


Fig. 5: Example images from our selected **VPR** benchmark datasets.

Each one of the five datasets brings along a different set of particularities and challenges:

a) **ZuBuD (Zurich Buildings Database)**: ZuBuD [34] contains 115 query and 1005 reference images of city buildings in Zurich. The high similarity of the buildings challenges the discretization capabilities of search methods.

b) **Oxford Buildings Dataset**: This dataset [33] consists of 55 query and 5008 reference images collected from Flickr by searching for Oxford landmarks. It has been manually annotated to generate a reliable ground truth.

c) **Paris Dataset**: The Paris Dataset [35] consists of 55 query and 6357 reference images collected from Flickr by searching for landmarks in Paris. Many shots have been taken at night, significantly complicating the matching.

d) **Holidays Dataset**: The Holidays dataset [17] is a set of 500 query and 991 reference images, many captured with high camera resolutions (3-6 megapixels). It includes a large variety of scene types (natural, man-made, etc.).

e) **KITTI VO / SLAM Evaluation 2012**: The popular benchmark dataset [36] for **VO** and **SLAM** approaches contains 6 loop closure sequences with ground truth trajectories. We computed an image matching ground truth using the provided trajectories and geometric verification.

³OpenCV library: www.opencv.org

⁴DBoW2 library: www.github.com/dorian3d/DBoW2

⁵HBST library: www.gitlab.com/srrg-software/srrg_hbst

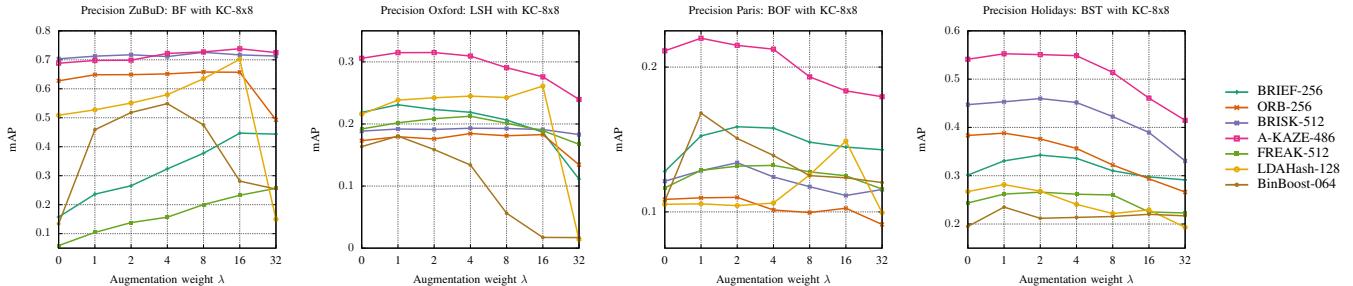


Fig. 6: Results for **KC**-8x8 with varying augmentation weight λ . A maximum of 1000 descriptors has been computed for each image.

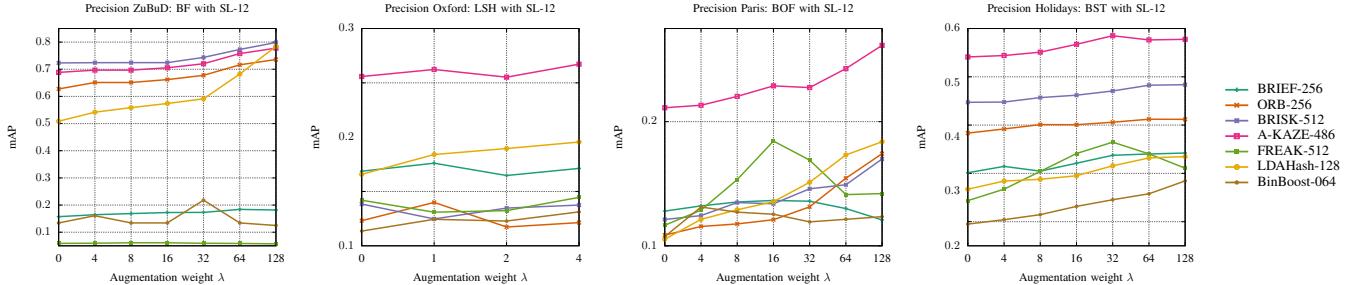


Fig. 7: Results for **SL**-12 with varying augmentation weight λ . A maximum of 1000 descriptors has been computed for each image.

E. Results

We first review our results for the standard image retrieval datasets ZuBuD, Oxford, Paris and Holidays, using a particular search method for each dataset. In each case we computed the **mAP**, using the public evaluation tool of [33]. In Fig. 6 and Fig. 7 we display the obtained **mAP** scores when augmenting descriptors with their keypoint coordinates (**KC** with $I_u = I_v = 8$) and semantic labels (**SL**). For each image, we computed a maximum of 1000 descriptors, sorted in descending order by their keypoints' response.

We evaluated the **mAP** scores for increasing augmentation weights λ . Note that for $\lambda = 0$ (i.e. cue is not considered) the top and their respective bottom plots start from the same **mAP** value. We mention the original number of bits for each evaluated descriptor type with a suffix (e.g. BRIEF-256).

Two descriptors are considered to match if their Hamming distance $L_H^\lambda(\mathbf{d}_*, \mathbf{d}'_*)$ lies within the threshold τ . For all datasets we set τ to 10% of the augmented descriptors size (e.g. $\tau = 27.2$ for BRIEF-256 with **KC**-8x8). Hence, τ grows as we increase λ , reducing precision in case noise is added. Since the **SL** augmentation results in only distances of 1 respectively 0 although the encoding length is 12 bits, we adjusted our threshold τ accordingly and additionally evaluated higher λ (i.e. 64 and 128 for **BF** in Fig. 7).

ZuBuD (BF): The keypoint coordinates turn out to be a highly beneficial cue when combined with **BF**. The significant precision drop at $\lambda = 4$ for BinBoost, and at $\lambda = 32$ for LDAHash and ORB marks the point at which the cues contribution \mathbf{b} saturates the descriptor \mathbf{d} : $L_H^\lambda(\mathbf{d}_*, \mathbf{d}'_*) \approx \lambda L_H(\mathbf{b}, \mathbf{b}') \gg L_H(\mathbf{d}, \mathbf{d}')$. In this case, relevant image information stored in \mathbf{d} is neglected and the comparison is based only on the cues. Clearly, $L_H(\mathbf{b}, \mathbf{b}')$ is not sufficiently descriptive anymore to find true matches within thousands of images. The smaller the used descriptor type, the earlier this happens. This phenomena can be observed in all datasets.

Considering the semantic labels, the gain in precision is still observable yet more restrained.

Oxford (LSH): The precision gain for both cue types is mild, yet an improvement can be observed for $\lambda = 1$ for every search method and descriptor type.

Paris (BOF): In the Paris dataset we observe clear precision peaks for several descriptor types. The **BOF** model profits from the additional information from our cues.

Holidays (BST): The employed **BST** approach benefits from our cues, yet only marginally. The Holidays dataset contains the most diverse images of all the datasets (e.g. from buildings to nature to human faces). In this challenging scenario our semantic labels are insufficient to provide significant additional information.

KITTI: For the evaluation on the KITTI dataset we built the image database incrementally for each approach. While processing the sequence of images, every image was first queried against the database of previous images and then added in a subsequent step, followed by a training phase. Clearly, this increases the computational demand for all methods considered as the database cannot be constructed in a single step. We processed every 10th image in a sequence, to avoid capturing many noisy descriptor versions.

In contrast to the previous datasets (Fig. 6 and Fig. 7), there are no rotated images in KITTI. Hence when revisiting a place, similar features are automatically detected in similar image regions. This clearly increases the utility of our keypoint coordinates cue, which is observable in our results.

Instead of the **mAP** we computed the complete **PR** curves for KITTI sequence 00. In Fig. 1 we display the resulting performance boost obtained by augmenting BRIEF-256 descriptors with our continuous and our selector cue. In Fig. 8 we display **PR** curves for $\lambda = 0$ respectively $\lambda = 16$ in comparison with each search method for the remaining 6 descriptor types.

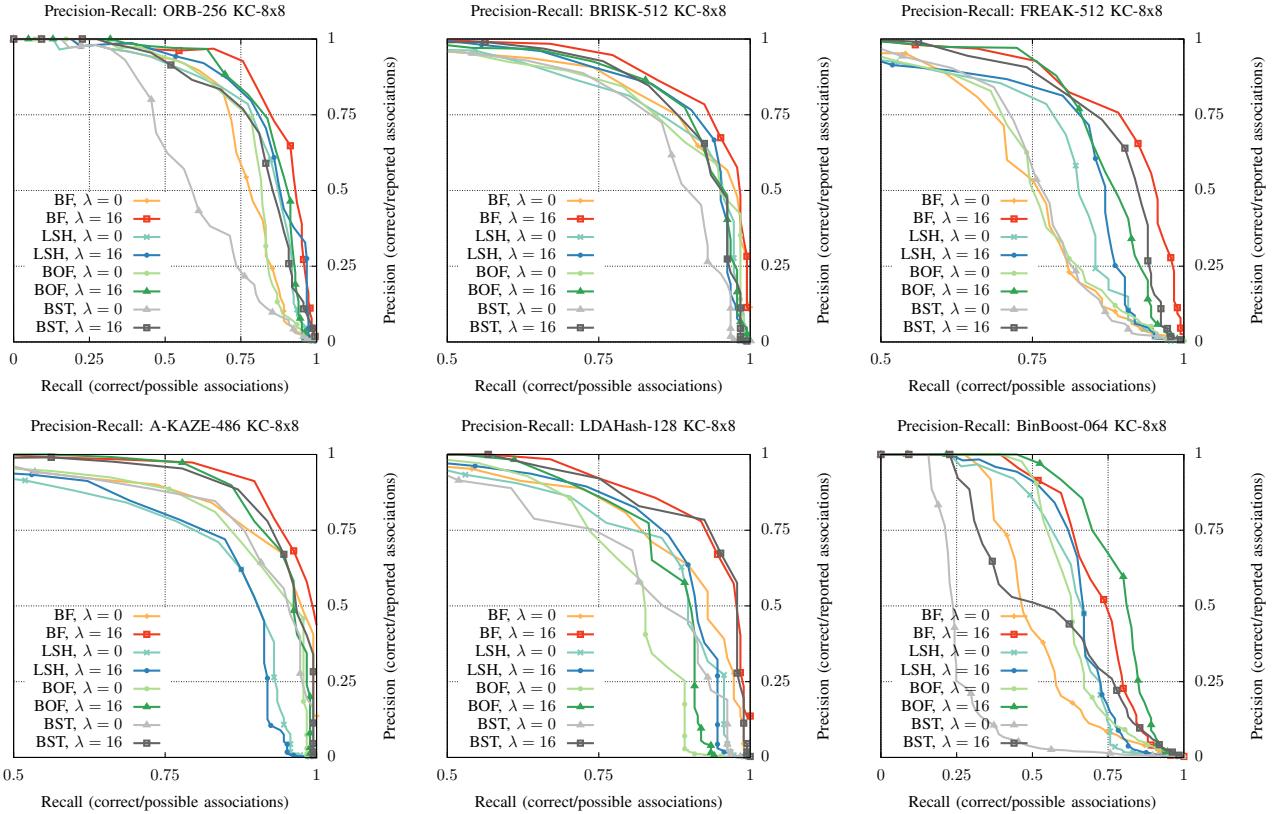


Fig. 8: PR analysis on KITTI sequence 00 for various descriptor types for the continuous example cue KC-8x8. For all search methods and descriptors we evaluated the cue weighting $\lambda = 0$ (cue ignored) and $\lambda = 16$. (see Fig. 1 for the PR curves of BRIEF).

As in the previous datasets, a clear improvement of the search precision is observable. The improvement however, is significantly more pronounced in KITTI as in the previous datasets. This is due to the mentioned missing rotated imagery and road bound image acquisition. Impressively, the evaluated **BST** approach (black) manages to reach almost **BF** (red) precision (FREAK, A-KAZE, LDAHash) when using our cues. The semantic cues do not manage to improve the performance as much as the keypoint coordinate cues in this case (Fig. 1). Yet, they have a positive effect on precision that is consistent with the results of the previous datasets.

In Tab. I we list the measured mean query processing times of the approaches considered in Fig. 1 and Fig. 8 on KITTI. We separately examined the specific augmentation weights $\lambda \in \{0, 1, 16, 32\}$ for the continuous augmentation case (**KC**).

Augmentation weight $\lambda = 0$				Augmentation weight $\lambda = 1$				
	BF	LSH	BOF	BF	LSH	BOF	BF	
BRIEF-256	2.032	0.950	0.153	0.002	3.141	1.015	0.207	0.002
ORB-256	2.689	1.133	0.170	0.001	3.046	1.087	0.190	0.002
BRISK-512	3.781	1.139	0.261	0.002	3.430	1.208	0.265	0.002
A-KAZE-486	3.517	1.078	0.259	0.002	3.711	1.096	0.281	0.002
FREAK-512	3.023	1.010	0.262	0.003	3.358	0.992	0.267	0.003
LDAHash-128	2.561	1.319	0.090	0.001	2.789	1.195	0.087	0.002
BinBoost-064	2.693	0.735	0.084	0.002	3.035	0.727	0.080	0.007

Augmentation weight $\lambda = 16$				Augmentation weight $\lambda = 32$				
	BF	LSH	BOF	BF	LSH	BOF	BF	
BRIEF-256	3.251	0.907	0.215	0.003	4.328	0.918	0.263	0.005
ORB-256	3.858	1.081	0.202	0.002	4.199	0.829	0.280	0.008
BRISK-512	3.296	0.911	0.313	0.004	4.040	0.929	0.365	0.011
A-KAZE-486	3.853	0.943	0.306	0.003	4.114	0.779	0.375	0.009
FREAK-512	3.146	0.824	0.354	0.006	3.749	0.884	0.433	0.011
LDAHash-128	2.802	0.711	0.112	0.002	3.025	0.723	0.133	0.008
BinBoost-064	3.005	1.280	0.086	0.009	3.273	2.917	0.111	0.009

TABLE I: Mean processing times \bar{t} (seconds) for differently weighted KC augmentations on KITTI sequence 00. We report the average \bar{t} over 10 runs.

In the case of $\lambda = 0$, the augmentation is not considered and hence the descriptor size is minimal. As expected, the processing time \bar{t} is the smallest compared to $\lambda > 0$ for most approaches. For $\lambda = 16$ and $\lambda = 32$ the processing times are still adequate for most search approaches.

V. CONCLUSIONS

In this paper, we presented approach that improves the precision for feature-based VPR by embedding continuous and selector cues into binary feature descriptors. Our approach is purely supplementary to the descriptor computation and the similarity search method. We implemented and evaluated our approach on several standard benchmark datasets and covered a vast number of state-of-the-art binary descriptor type and search method combinations. Our results suggest that our strategy is effective and increases VPR precision, regardless of the descriptor type and search method.

REFERENCES

- [1] A. Torralba, K. P. Murphy, W. T. Freeman, M. A. Rubin, *et al.*, “Context-based vision system for place and object recognition..,” in *Proc. of the Int. Conf. on Computer Vision (ICCV)*, vol. 3, pp. 273–280, 2003. [1](#)
- [2] D. Gálvez-López and J. D. Tardós, “Bags of binary words for fast place recognition in image sequences,” *IEEE Trans. on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012. [1](#), [2](#), [5](#)
- [3] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, “Visual place recognition: A survey,” *IEEE Trans. on Robotics*, vol. 32, no. 1, pp. 1–19, 2016. [1](#)
- [4] C. Zhu, “Place recognition: An Overview of Vision Perspective,” *CoRR*, vol. abs/1707.03470, 2017. [1](#)
- [5] A. Oliva and A. Torralba, “Modeling the shape of the scene: A holistic representation of the spatial envelope,” *Int. Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001. [1](#)
- [6] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004. [1](#), [2](#), [3](#), [4](#)
- [7] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2005. [1](#)
- [8] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” in *Proc. of the Eur. Conf. on Computer Vision (ECCV)*, 2010. [1](#), [2](#), [3](#), [4](#)
- [9] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in *Proc. of the Int. Conf. on Computer Vision (ICCV)*, 2011. [1](#), [2](#), [3](#), [4](#)
- [10] S. Leutenegger, M. Chli, and R. Y. Siegwart, “BRISK: Binary robust invariant scalable keypoints,” in *Proc. of the Int. Conf. on Computer Vision (ICCV)*, 2011. [1](#), [2](#), [3](#), [4](#)
- [11] P. F. Alcantarilla and T. Solutions, “Fast explicit diffusion for accelerated features in nonlinear scale spaces,” *IEEE Trans. Patt. Anal. Mach. Intelligence*, 2011. [1](#), [3](#), [4](#)
- [12] A. Alahi, R. Ortiz, and P. Vandergheynst, “FREAK: Fast retina keypoint,” in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012. [1](#), [2](#), [3](#), [4](#)
- [13] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua, “Ldahash: Improved matching with smaller descriptors,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 66–78, 2012. [1](#), [4](#)
- [14] T. Trzcinski, M. Christoudias, P. Fua, and V. Lepetit, “Boosting binary keypoint descriptors,” in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 2874–2881, 2013. [1](#), [5](#)
- [15] V. Balntas, L. Tang, and K. Mikolajczyk, “BOLD - Binary online learned descriptor for efficient image matching,” in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 2367–2375, 2015. [1](#)
- [16] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li, “Multi-probe LSH: efficient indexing for high-dimensional similarity search,” in *Proc. of the 33rd Int. Conf. on Very large data bases*, 2007. [1](#), [2](#)
- [17] H. Jegou, M. Douze, and C. Schmid, “Hamming embedding and weak geometric consistency for large scale image search,” in *Proc. of the Eur. Conf. on Computer Vision (ECCV)*, pp. 304–317, 2008. [1](#), [2](#), [5](#)
- [18] B. Sankaran, S. Ramalingam, and Y. Taguchi, “Parameter learning for improving binary descriptor matching,” in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 4892–4897, 2016. [1](#), [2](#)
- [19] K. Mikolajczyk and J. Matas, “Improving descriptors for fast tree matching by optimal linear projection,” in *Proc. of the Int. Conf. on Computer Vision (ICCV)*, pp. 1–8, 2007. [1](#), [2](#)
- [20] T. Harada, H. Nakayama, and Y. Kuniyoshi, “Improving local descriptors by embedding global and local spatial information,” in *Proc. of the Eur. Conf. on Computer Vision (ECCV)*, pp. 736–749, 2010. [1](#), [2](#)
- [21] M. Kottman, “Improving binary feature descriptors using spatial structure,” *Information Sciences and Technologies*, vol. 5, no. 2, p. 32, 2013. [1](#), [2](#)
- [22] J. Wang, X. Wang, X. Yang, and A. Zhao, “CS-FREAK: an improved binary descriptor,” in *Chinese Conf. on Image and Graphics Tech.*, pp. 129–136, 2014. [1](#), [2](#)
- [23] X. Xiao, S. He, Y. Guo, M. Lu, and J. Zhang, “BAG: A Binary Descriptor for RGB-D Images Combining Appearance and Geometric Cues,” in *Int. Conf. on Cognitive Systems and Signal Processing*, pp. 65–73, 2016. [1](#), [2](#)
- [24] R. W. Hamming, “Error detecting and error correcting codes,” *Bell Labs Technical Journal*, 1950. [1](#), [2](#)
- [25] S. Madeo and M. Bober, “Fast, compact and discriminative: Evaluation of binary descriptors for mobile applications,” *IEEE Transactions on Multimedia*, 2016. [2](#)
- [26] J. Sivic and A. Zisserman, “Video google: A text retrieval approach to object matching in videos,” p. 1470, 2003. [2](#)
- [27] D. Schlegel and G. Grisetti, “HBST: A Hamming Distance embedding Binary Search Tree for Feature-based Visual Place Recognition,” *IEEE Robotics and Automation Letters (RA-L)*, 2018. [2](#), [5](#)
- [28] R. Mur-Artal and J. D. Tardós, “ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras,” *IEEE Trans. on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017. [3](#)
- [29] T. Pire, T. Fischer, J. Civera, P. De Cristóforis, and J. J. Berlles, “S-PTAM: Stereo Parallel Tracking and Mapping,” in *Journ. of Rob. & Aut. Systems*, vol. 93, pp. 27–42, 2017. [3](#)
- [30] D. Schlegel, M. Colosi, and G. Grisetti, “ProSLAM: Graph SLAM from a Programmer’s Perspective,” *CoRR*, vol. abs/1709.04377, 2017. [3](#)
- [31] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” in *Proc. of the Eur. Conf. on Computer Vision (ECCV)*, 2006. [3](#)
- [32] V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation,” *CoRR*, vol. abs/1511.00561, 2015. [4](#)
- [33] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, “Object retrieval with large vocabularies and fast spatial matching,” in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8, 2007. [5](#), [6](#)
- [34] H. Shao, T. Svoboda, and L. Van Gool, “ZuBuD - Zurich buildings database for image based recognition,” *Computer Vision Lab, ETHZ, Tech. Rep.*, vol. 260, no. 20, p. 6, 2003. [5](#)
- [35] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, “Lost in quantization: Improving particular object retrieval in large scale image databases,” in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008. [5](#)
- [36] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012. [5](#)