Pareto-optimal Dictionaries for Signatures*

Michael Calonder, Vincent Lepetit, Pascal Fua
CVLab, EPFL, Lausanne, Switzerland
e-mail: `firstname.lastname@epfl.ch`

Abstract

We present an effective method to optimize over the parameters of an image patch descriptor to obtain one that is computationally more efficient while maintaining a high recognition rate. We formulate the optimization problem in a multi-objective manner, which balances two conflicting goals while removing the need for traditional weighting coefficients. To this end we introduce the Pareto efficiency *criterion, which helps finding solutions that increase one objective without decreasing the other. Despite the vast size of the search space, we show how a state-of-the-art Genetic Algorithm can be tailored to find good solutions.*

Not only does the resulting descriptor perform better than state-of-the-art ones, but our approach is of broader significance as optimization problems with balanced goals are often encountered in Computer Vision.

1. Introduction

Many vision applications rely on local image descriptors to establish correspondences between images. SIFT [9] has become the *de facto* reference, and fast implementations now exist and run in real-time on relatively slow devices such as cell phones [2, 13]. This, however, is not the end of the story because real-time performance is only obtained for limited numbers of points and requires most of the CPU, whereas a truly useful application should be able to run in the background.

Recently, an approach with similar accuracy but substantially decreased computational requirements has been introduced [3, 4]. It involves training a fast classifier [12] offline to recognize keypoints from a *dictionary* and taking its response to previously unseen keypoints as description vectors, or *Signatures*, which can be matched by nearest neighbor search. In the first version of that approach, the dimension of the Signatures was that of the dictionary, about 500 elements, which is too large for maximal efficiency. It was then shown that the dimensionality could be dropped to 176 by exploiting the sparsity of the Signatures and projecting them in a lower dimensional space using Random Orthoprojections. This resulted in run-times about 10 times as fast as those of SURF [2], with comparable matching accuracy.

In this paper, we replace the Random Orthoprojections by a Genetic Algorithm (GA) that selects the best subset of keypoints from the dictionary and balances the desired recognition rate against Signature size and therefore algorithmic complexity. This way, not only can we almost double the speed for the same matching performance of the already fast method in [4] but we can also optimally trade some matching performance against additional speed if we want to handle more points or run on a less capable device.

Selecting the best subset of keypoints is a large combinatorial problem and requires powerful optimization techniques to explore the search space effectively. In this context, GAs can be designed to find good "Pareto optimal" solutions, that is solutions to a multi-criterion optimization problem such that any improvement on one criterion would inevitably render them worse in another criterion. It follows that we do not have to weigh the different optimization goals, which is a recurrent issue in Computer Vision.

For the above-mentioned reasons we believe that our approach tailoring a state-of-the-art GA [15] to the specific needs of this problem is of interest for many other, similar problems in Computer Vision. For example, [14] and [8] previously described the optimization of local image descriptors but [14] optimized only on the recognition performances whereas the method introduced in [8] is sub-optimal by design to avoid the combinatorial explosion.

In the remainder of the paper we start by reviewing related work. We then formalize our problem, and show how to solve it with a Genetic Algorithm. Finally we compare the optimized Signatures obtained from the GA against earlier ones and SURF. We demonstrate substantial improvement in both cases.

2. Related Work

Many of the local image descriptors—SIFT, SURF, and others—were hand-crafted and, as such, there is no guar-

*This work has been supported in part by the Swiss National Science Foundation.

antee that there is no set of parameters that would make them work better. A notable exception are the descriptors presented in [14]. This is the method most similar to what we propose that we know of, however it only optimizes the descriptors' parameters to maximize the recognition performance. In this work, we explicitly look for solutions that balance recognition performance against descriptor complexity.

A different approach is to look for projections that reduce the dimensionality of the descriptor. PCA is a popular choice, but there is no proof of optimality either. Therefore [8] uses a variant of LDA to build a sub-optimal projection but, because of the combinatorial nature of the problem, the optimization proceeds one direction only at a time, an can get stuck in a local minimum. By contrast, by means of the Genetic Algorithm, we are able to perform a full-scale optimization on this rather complex problem.

3. Formulation

We first summarize the Compact Signatures method of [3] and formalize the dictionary selection problem that aims at making these Signatures computationally effective while preserving their recognition performance. We then show how this problem can naturally be set up in the Pareto efficiency framework and explain in Section 4 how to solve it using a widely-applicable multi-objective Genetic Algorithm.

3.1. Signatures for Keypoint Matching

The Signatures of [3, 4] are computed by first training offline a Fern classifier [11] to recognize a set of keypoints, which we call a *dictionary*. The classifier response to another keypoint *not* in the dictionary is a set of probabilities, expressing the similarity of that keypoint to all of the dictionary points. It is characteristic and stable under viewpoint variations, lighting changes, and noise. It can therefore be treated as a descriptor for arbitrary keypoints and can be matched by simple nearest neighbor search in descriptor space.

More precisely, as sketched in Figure 1, a Signature $\mathbf{s}' \in \mathbb{R}^N$ for an image patch $\mathbf{p}$ is computed as

$$\mathbf{s}'(\mathbf{p}) = \sum_{1 \le i \le J} \mathbf{t}_i(\mathbf{p}) \qquad (1)$$

where J is the number of Fern classifiers [11] and the $\mathbf{t}_i \in \mathbb{R}^N$ denote the leaves of the Ferns indexed by $\mathbf{p}$. N is the size of the dictionary, and each coordinate of the $\mathbf{t}_i$ vectors corresponds to a keypoint in the dictionary, whose keypoints were picked from an arbitrary image.

Because Signatures computed that way are long vectors and thus not efficient to match, [4] uses a simple Random

The only constraint being that the image exhibits enough "structure".

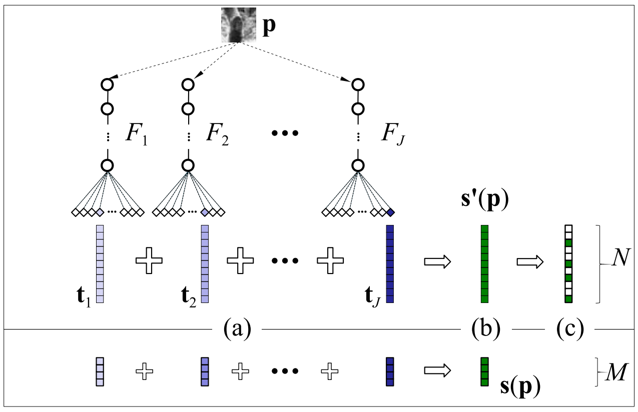


Figure 1. Keypoint Signature computation. **Top:** (a) The patch $\mathbf{p}$ surrounding the keypoint is dropped through all ferns F_i, indexing into J $\mathbf{t}_i$ vectors. (b) All $\mathbf{t}_i$ are summed up to get the classifier response $\mathbf{r}(\mathbf{p})$. (c) Thresholding yields a Sparse Signature. **Bottom:** Summation of J compressed vectors which is all that needs to be carried out at runtime.

Orthoprojection to reduce the dimensionality of the $\mathbf{t}_i$ vectors and therefore of the resulting Signature $\mathbf{s} \in \mathbb{R}^M$. By contrast, in this paper, we show how to select the best subset of keypoints to create a short dictionary with good recognition performance.

3.2. Creating a Dictionary

Creating a dictionary for training the classifier is inherently difficult: A typical dictionary contains $N \approx 500$ elements and exhaustive search would therefore have to evaluate $2^N \approx 10^{150}$ solutions, which is absurdly large.

However, in combinatorial optimization problems heuristics often work remarkably well. Assuming that the dictionary's keypoints can be treated independently, we could use a simple greedy algorithm that needs to evaluate "only" about N^2 solutions and pick the best one. Unfortunately, as will be shown in the results section, keypoints are interdependent and the resulting solutions are poor. We therefore use a Genetic Algorithm to perform the exploration. Also, it is important to note that the recognition rate relates in a non-monotonic way to the number of dictionary elements and hence varying that parameter is insufficient to find a good set of elements.

The search for a good dictionary is made even more difficult by the fact that we need to balance out two competitive goals, that is the size of the dictionary and the recognition performance. To avoid having to weight these conflicting goals, we optimize the Pareto efficiency, as described below.

3.3. Problem Formalization

From a large initial dictionary containing N keypoints, we seek to select a subset of keypoints that is optimal w.r.t.

our two competitive goals. A solution $\mathbf{x}$ is defined as a N-vector of 0 and 1 values indicating which of the dictionary's elements are active.

Our problem can be formulated as seeking to solve the bi-objective optimization problem:

$$\begin{cases} \max f_1(\mathbf{x}) \\ \min f_2(\mathbf{x}) \end{cases}, \tag{2}$$

where $f_1(\mathbf{x})$ measures the recognition rate for dictionary $\mathbf{x}$, and $f_2(\mathbf{x})$ is the number of active keypoints in $\mathbf{x}$, or the number of coordinates set to 1.

To evaluate $f_1(\mathbf{x})$ we use a training set $\mathcal{T}$ made of matching pairs of points in two images. In practice we create it by extracting several keypoints from a training image and computing random views of them by warping it. $f_1(\mathbf{x})$ is then taken to be the recognition rate when matching the points in $\mathcal{T}$ to their nearest neighbors, based on the Signatures computed using only the active keypoints in $\mathbf{x}$.

Traditional algorithms minimize normalized objectives. We therefore define $\bar{f}_1(\mathbf{x}) = -f_1(\mathbf{x})$ and $\bar{f}_2(\mathbf{x}) = \frac{1}{N} f_2(\mathbf{x})$ and reformulate the problem as

$$\begin{cases} \min \bar{f}_1(\mathbf{x}) \\ \min \bar{f}_2(\mathbf{x}) \end{cases}, \tag{3}$$

which is equivalent to Eq. 2.

3.4. Pareto Optimality

In the single-objective case, it is straightforward to compare the quality of two solutions based on the objective function, whereas in the above bi-objective problem, no such natural ordering exists. As illustrated by Figure 2, only a partial ordering is possible, and if

$$\begin{array}{ll} \bar{f}_1(\mathbf{x}_1) \le \bar{f}_1(\mathbf{x}_2) \text{ and } \bar{f}_2(\mathbf{x}_1) < \bar{f}_2(\mathbf{x}_2) & \text{or} \\ \bar{f}_1(\mathbf{x}_1) < \bar{f}_1(\mathbf{x}_2) \text{ and } \bar{f}_2(\mathbf{x}_1) \le \bar{f}_2(\mathbf{x}_2) & , \end{array} \tag{4}$$

$\mathbf{x}_1$ is said to *dominate* $\mathbf{x}_2$, or $\mathbf{x}_1 \succ \mathbf{x}_2$.

A solution $\mathbf{x}$ is said to be *Pareto optimal* if there is no other solution $\mathbf{x}'$ that dominates $\mathbf{x}$. The *Pareto set* is the set that of all non-dominated solutions.

4. Optimization

A Genetic Algorithm (GA) is an iterative, stochastic optimization method that works on a *population* of solutions [10]. Good introductory texts can be found in [5, 7, 10], for example. Inspired by evolutionary biology, GAs apply at each iteration operations such as mutation, crossover, and selection to the population, driving the solutions towards a optimum defined by the objective functions. Specifically, *variation operators* work on a solution $\mathbf{x}$ by flipping one or more bits or copying parts from or to another solution. By contrast, *selection* defines which of the solutions survive.

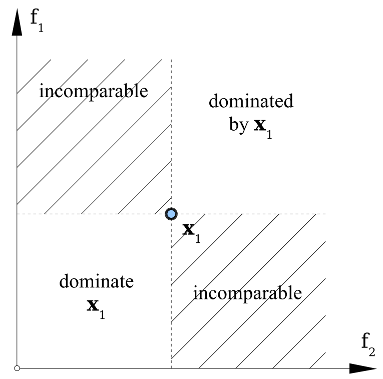


Figure 2. Dominance concept. In a multi-objective problem, two solutions can be incomparable, as opposed to single-objective problems.

Parameter	Value
Population size $\|P\|$	200
Mutation probability p_m	0.2
Recombination probability p_r	0.7
Independent bit flip probability	$3/\|P\| = 0.015$
Number of iterations N_{it}	445

Table 1. GA parameters. Typical values for the population size are around 100. However, the huge search space reduces the solution density dramatically and promotes genetic drift, which both justify an increased number of solutions.

4.1. Variation Operators

Variation encompasses the mutation and crossover operations. Mutation accounts for small variations that occur naturally and at random where crossover defines how two good solutions should be combined to form two other, even better ones. Here we briefly detail how a solution $\mathbf{x}$ is affected by these operators.

Mutation operator. We experimented with two mutation operators, one that independently flips a coordinate and one that flips a 1 and a 0 and hence preserves the solution size. Experimentally we found that the first one performs slightly better.

Crossover operator. The operator takes two solutions and recombines them by exchanging blocks of bits, yielding two new solutions. This is often called two-point crossover.

4.2. Solution Selection

Given the variation operator described above, a critical component of GAs is the selection of good solutions for the next iteration. The ultimate goal of this selection is to make the population P of solutions converge towards the *Pareto* *set*. Most of the successful methods to-date [6, 1, 16, 15] achieve this by computing a *fitness* measure $F(\mathbf{x})$ for each $\mathbf{x} \in P$ and differ merely in the way F is defined. Note that F maps the two objective values to a scalar which is to be maximized.

For our problem, we have tried the two probably most successful selectors of the past years, SPEA2 [16] and IBEA [16], and found superior performance for the latter, which we briefly summarize here.

Given the population size α and the maximum number of iterations N_{it}, IBEA outputs an approximation A to the Pareto set. This involves the following steps:

1. **Initialization.** We initialize the population with 200 solutions of length $N = 500$. The number of coordinates set to 1, or active keypoints, is uniformly distributed over $[10, 500]$ in order to avoid a bias on the length.

2. **Fitness Assignment.** Compute $F(\mathbf{x}_i)$ for all $\mathbf{x}_i \in P$.

3. **Environmental selection.** Remove $\mathbf{x}^\star = \arg\min_{\mathbf{x} \in P} F(\mathbf{x}_i)$, recompute the fitness values and repeat until the population is reduced to α solutions.

4. **Termination.** If N_{it} is reached, take A to be all non-dominated solutions in the current population and stop.

5. **Mating selection.** Choose solutions for the mating pool P' using binary tournament selection with replacement on P.

6. **Variation.** Apply crossover and mutation operators to P' and add the resulting offspring to P. Go to step 2.

IBEA computes the fitness measure $F(\cdot)$ based on what is called the "binary additive ϵ-indicator", usually denoted I_ϵ^+ [15]. In our case, and as illustrated by Fig. 3, I_ϵ^+ reduces to a function that measures the relative quality between two solutions, as the amount of translation needed for the first solution to be as good as the second one. This can be written as

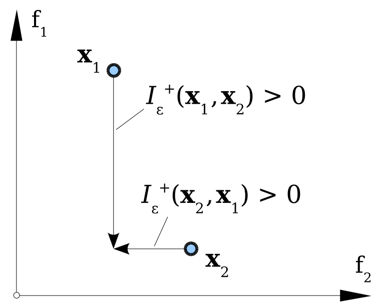
$$\begin{aligned} & I_\epsilon^+(\mathbf{x}_1, \mathbf{x}_2) = \epsilon, \text{ s.t.} \\ & \quad \forall i \,:\, \bar{f}_i(\mathbf{x}_1) - \epsilon \le \bar{f}_i(\mathbf{x}_2) \text{ and} \\ & \quad \nexists\, \epsilon',\ 0 < \epsilon' < \epsilon,\ \text{ s.t. } \forall i \,:\, \bar{f}_i(\mathbf{x}_1) - \epsilon' \le \bar{f}_i(\mathbf{x}_2)\,. \end{aligned}$$

Then the fitness $F(\mathbf{x})$ for solution $\mathbf{x}$ in a population P reads

$$F(\mathbf{x}) = \sum_{\mathbf{x}' \in P \setminus \{\mathbf{x}\}} -\exp\left(-\frac{I_\epsilon^+(\mathbf{x}', \mathbf{x})}{\kappa}\right) \tag{5}$$

with $\kappa > 0$ a scaling factor.

A refined version of the algorithm allows to set $\kappa = 0.05$ for all optimization problems by requiring $\forall \mathbf{x}_i, \mathbf{x}_j \in P$: $I_\epsilon^+(\mathbf{x}_i, \mathbf{x}_j) \in [-1, 1]$, rendering κ independent of the problem at hand. $\kappa = 0.05$ was found via benchmark problems [15].



f1
f2
x1
x2
$I_\varepsilon^+(\mathbf{x}_1,\mathbf{x}_2) > 0$
$I_\varepsilon^+(\mathbf{x}_2,\mathbf{x}_1) > 0$

Figure 3. Illustration of the ϵ-indicator I_ϵ^+. $I_\epsilon^+(\mathbf{x}_1, \mathbf{x}_2)$ measures how much $\mathbf{x}_1$ needs to be translated in its worst objective so that it gets as good as $\mathbf{x}_2$. Also, $I_\epsilon^+(\mathbf{x}_1, \mathbf{x}_2) \neq I_\epsilon^+(\mathbf{x}_2, \mathbf{x}_1)$ in general.

Furthermore, it is easy to show [15] that F based on I_ϵ^+ satisfies $\mathbf{x}_i \succ \mathbf{x}_j \Rightarrow F(\mathbf{x}_i) > F(\mathbf{x}_j)$, which is called Pareto dominance relation, and is important for correct operation of any multi-objective GA.

We experimented with the GA's parameters and found no particular sensitivity to any of them. The final values we used in our experiments are given in Table 4.1.

5. Results

To assess matching performance, we use the same two publicly available datasets as in the Signature paper [4]. These datasets are called Wall, and Fountain. The Wall scene is planar and the relationship between two images in the database can be expressed by a homography. By constrast the Fountain scene is fully three-dimensional. For illustration, the datasets are shown in Figure 4.

5.1. Convergence of the GA

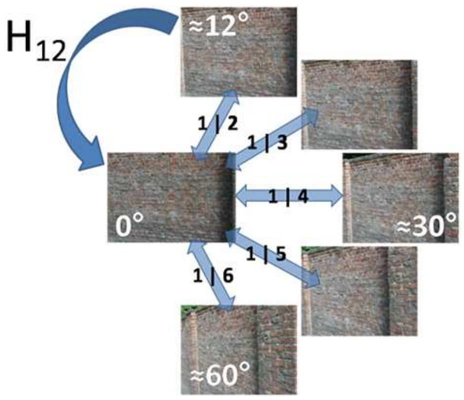
One weakness of GAs is that there is no generic stopping criterion which requires us to define one for our application. We consider that the algorithm has converged when the mean objective values f_1 and f_2 have changed no more than 0.1% over the last 10 iterations. Under this criterion, we ran the simulation for 445 generations.

The resulting solutions exhibit a structure that is highly unlikely under the assumption that the bits were chosen at random. Because a random solution is the most general or the least adapted one to the problem, comparing the solutions's structure to the that of random solutions is a good

To be accurate, this version of the algorithm is called "Adaptive IBEA" although adaptive behavior is usually implicit when speaking of IBEA, as it is in this paper.

http://www.robots.ox.ac.uk/~vgg/research/affine

http://cvlab.epfl.ch/~strecha/multiview



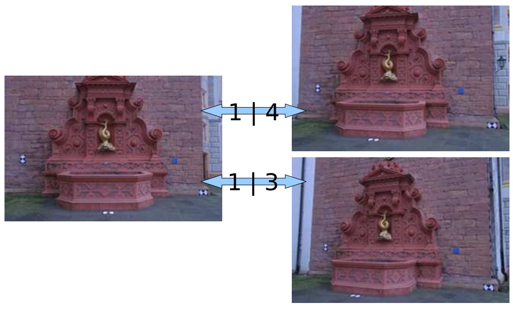


Figure 4. Datasets used for evaluation. **Left:** 6 Wall datasets. All homographies H_{1i} for $2 \leq i \leq 6$ are known, H_{12} is indicated for illustration. **Right:** 2 Fountain datasets. Ground-truth point-wise known.

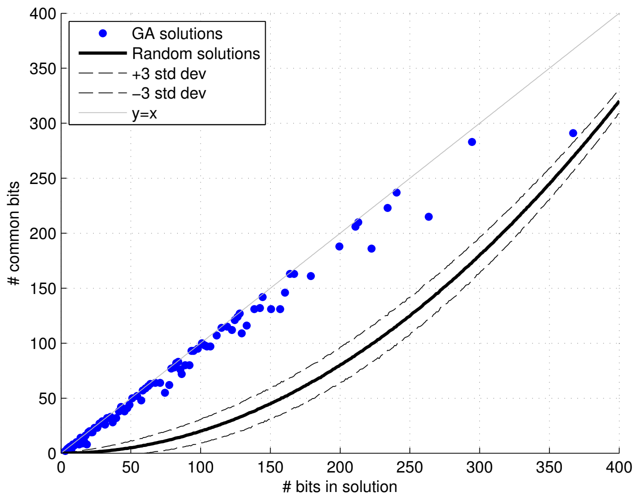


Figure 5. Compares the number of *common* bits for GA ('●') and random solutions. The thick line shows the mean number of common bits in random solutions for a given size and the dashed lines mark $\pm$ 3 standard deviations. The gray line represents $y = x$ (i.e. all bits identical). To find the number of common bits for a GA solution $\mathbf{x}$, we take the solution $\mathbf{x}'$ with the same (or if not possible, a similar) number of bits and compute $\mathbf{x}^\top\mathbf{x}'$. We see that GA solutions are i) highly non-random and ii) for a given number of bits very similar, indicating the existence of stable keypoint groups in the dictionary.

indicator for the quality of the solution for the present problem, as shown in Figure 5.

5.2. Compressibility of GA Solutions

As earlier work [3, 4] showed, sparsity in descriptors is desirable because it allows compression by a Random Orthoprojection (ROP) without a significant loss of recognition rate. This even holds if for signatures that are only sparse in *some* space of the same dimension.

Vice versa, we can assess the compressibility of a solution by projecting it to a lower dimensional space with a fixed number of dimensions, which we choose to be 104. This way we can compress GA solutions with dimension $d_i > 104$ as well as random solutions of the same dimension with an ROP and compare the two in terms of recognition rate. The solution with the higher recognition rate must be sparser in its original representation and is therefore better compressible. The result is shown in Figure 6 and suggests that in all cases GA-induced Signatures compress better than Compact Signatures.

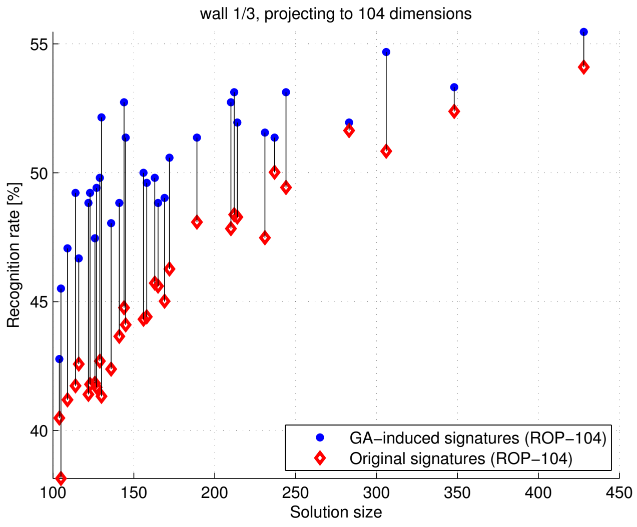


Figure 6. Compressibility of GA solutions. GA-induced Signatures are significantly better compressible than Compact Signatures (ROP).

5.3. Random and Greedy Approach

In this section we compare the GA-optimized Signatures against those obtained by a random algorithm and by a

As random solutions merely are instances of Compact Signatures, averaging over them provides us with a good estimate of the Compact Signatures' performance.

greedy one. These competing methods are:

- Random: Generate a pre-defined number of random solutions $\mathbf{x}_i$, evaluate them and keep the best for the given dictionary size.
- Bottom-up: Start with $\mathbf{x}_0 = (0, 0, \ldots, 0)^\top$ and add the element that improves the solution the most, yielding $\mathbf{x}_1$. Iterate until the desired number of elements is reached.
- Top-down: Start with $\mathbf{x}_0 = (1, 1, \ldots, 1)^\top$ and remove the element that decreases the performance the least, yielding $\mathbf{x}_1$. Iterate until the desired number of elements is reached.

Note that the random approach disregards any group structure in the selected elements whereas the latter two approaches consider a very limited neighborhood in the search space. The latter two approaches compute the recognition rate on the same test set $\mathcal{T}$ as described in Section 3.1.

All of these approaches, however, are clearly outperformed by the GA, as shown in Figure 7.

5.4. Comparison to Compact Signatures

Here we compare the GA and the three simple approaches to the dictionary selection problem with the ROP approach proposed in [4], as shown in Figure 7. Note that

- down-projecting a 500-D Signature to 172-D by a ROP does only slightly better than selecting an arbitrary set of 172 keypoints from the dictionary. This should not be surprising since a random selection can always be written as a random orthogonal projection.
- the greedy top-down approach can reduce the descriptor length by 12%.
- the GA identifies by far the best set of dictionary elements, reducing the descriptor length by up to 50%. We attribute this behavior to the global design of the GA.

For instance, the top row of Figure 7 shows that a ROP projecting into 172 dimensions yields the same recognition rate as a GA solution with 104 dimensions. Or in terms of efficiency, the GA reduced the memory consumption of the underlying Fern classifier and the Signatures themselves by 41%, while the CPU time for matching drops by $1 - (1 - .41)^2 = 65\%$ w.r.t. the Compact Signatures, as $t_{\text{match}} \propto M^2$.

The GA result quantifies the trade-off between the recognition rate and the Signature length in an optimal way. Thus, from an end-user perspective, the operation point can be chosen according to the application or the available computational resources of the problem at hand.

Also, non-GA methods yield mostly suboptimal solutions under the above Pareto criterion, that is, most of the solutions are dominated, as shown in Figure 8.

We show no errorbars on the *random* curve as the variance is very small, typically below 0.5% for all M.

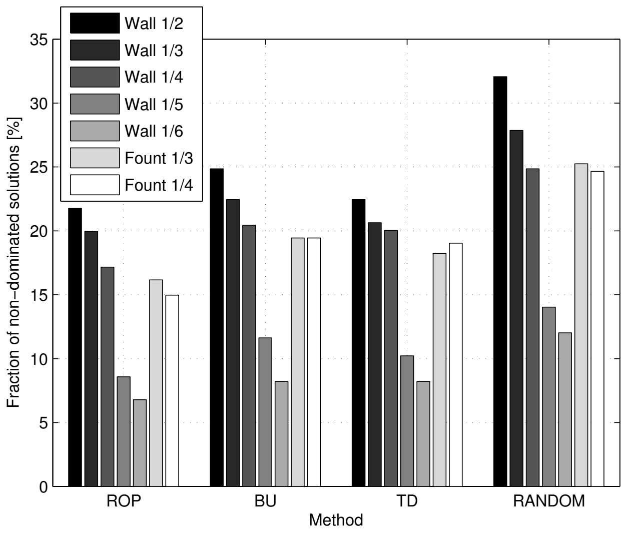


Figure 8. Percentage of *non*-dominated solutions for different methods, test sets and test cases. Note that all of the values are far below the value of the GA, which is not shown as it is always 100% by construction. Also, the number of dominated solutions shows a strong dependence on the difficulty of the test set.

5.5. Comparison to SURF

Recent work showed comparable recognition rates for Signatures and SURF [4] on different data sets and also across all view-points. Both methods select the match for a given keypoint as its nearest neighbor in descriptor space.

From a practical point of view, comparing the recognition rate may seem sufficient. However, this measure does not account for the robustness of the method, which comes into play when using approximate Neareast Neighbor schemes and we define it as the ordering that the respective descriptor imposes on the points in the descriptor space. To evaluate this measure, we introduce a ROC plot showing the recognition rate versus the fraction of candidate keypoints considered for matching. This is depicted in Figure 9 for a typical real-world test data set of intermediate difficulty. We see that the descriptor space induced by GA-optimized Signatures exhibits a much more robust ordering compared to that of SURF.

5.6. Selected Dictionary Elements

Visually inspecting the GA-selected dictionary is interesting although interpreting the selected elements is very

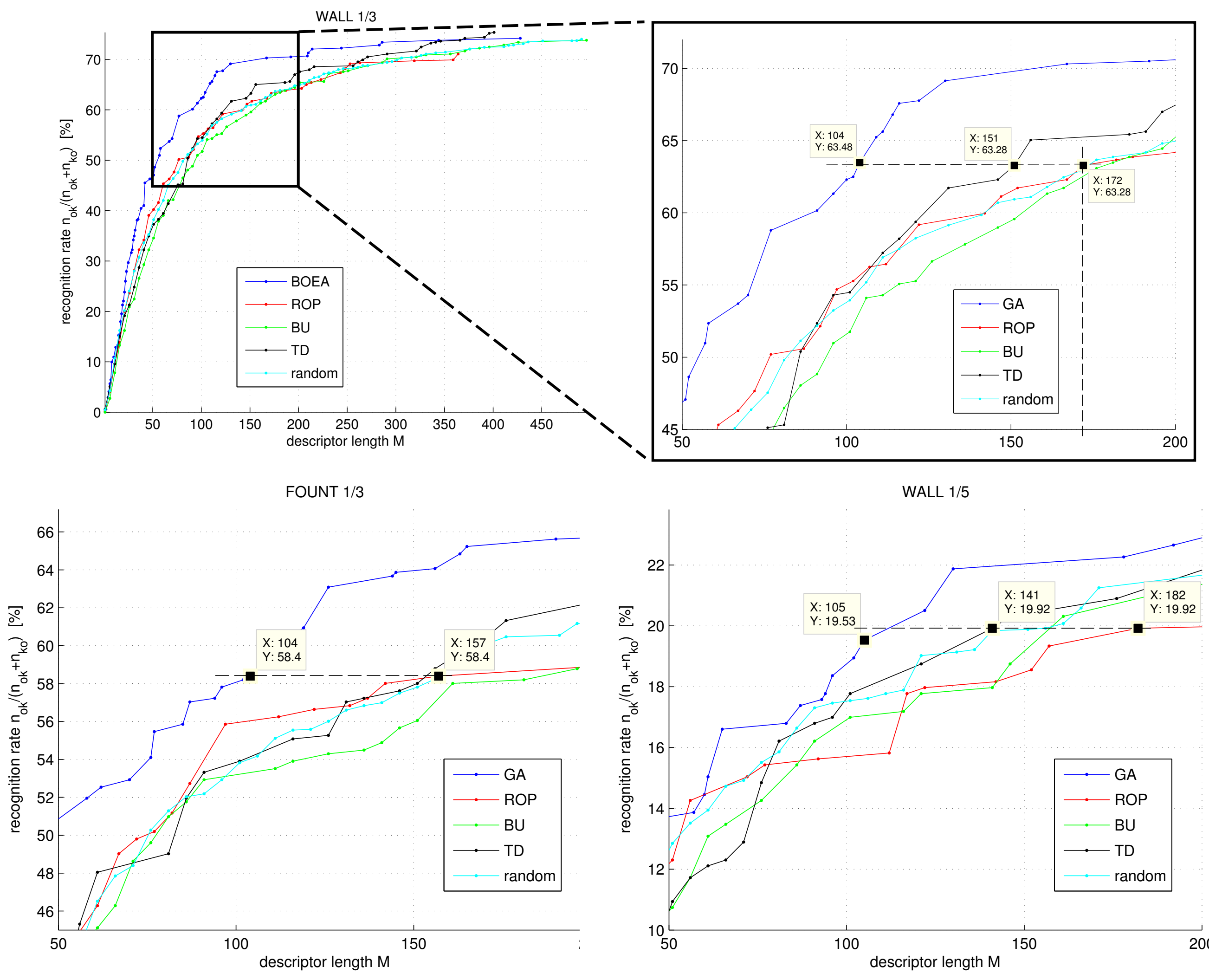


Figure 7. Comparison of five different dictionary selection methods, on three datasets: top, bottom left, bottom right (in order of increasing difficulty). Methods are random (Random), TD (top-down), BU (bottom-up), GA (Genetic Algorithm), and ROP (Compact Signatures based on a ROP). In the top right graph, the dashed vertical line is at $x \approx 176$, which is the dimensionality of the originally proposed descriptor [4] and intersects with the ROP curve. From that point, a horizontal line extends to the left where it intersects the GA curve. Clearly, the recognition rate is the same while the dimensionality of the GA is much lower. The analogous argumentation is true for the two datasets in the bottom row.

difficult as the classifier is highly non-linear. Figure 10 shows both the image source and the selected dictionary for an optimal solution of length 104. Note that each patch has its own distribution of bright and dark areas. We believe this to be an important aspect of a good dictionary because the *combination* of these keypoints is what defines the classifier's efficiency. Also the intuition that redundancy in the patch appearances should be avoided is confirmed.

6. Conclusion

We demonstrated the effectiveness of a Genetic Algorithm approach to balancing the two conflicting requirements of an algorithm designed to match feature points, namely high matching rates versus low computational requirements. Even though the search space is huge, we were able to tailor the Genetic Algorithm so that the resulting algorithm clearly outperforms its un-optimized predecessors over the whole operating range.

This is an important result because these type of compromises have to be made in a wide range of Computer Vision algorithms and, in future work, we intend to generalize our approach to other problems in that class.

We are commited to release the source code for both our implementation of the Compact Signatures and the Genetic Algorithm.

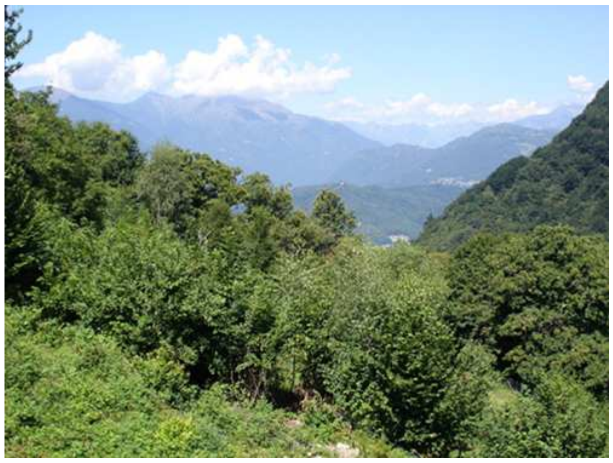
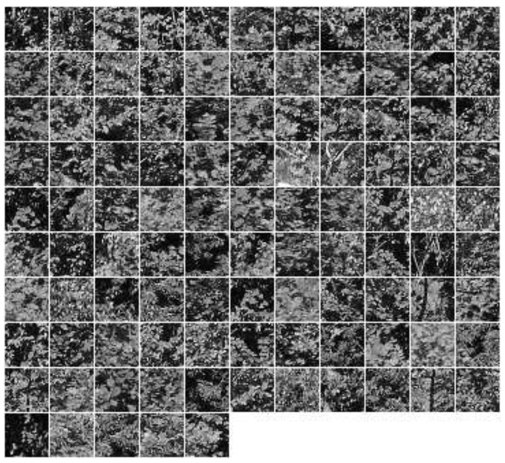



Figure 10. Left: The image that served as dictionary source for both the GA and the Compact Signatures [4]. Right: 104 patches the GA selected.

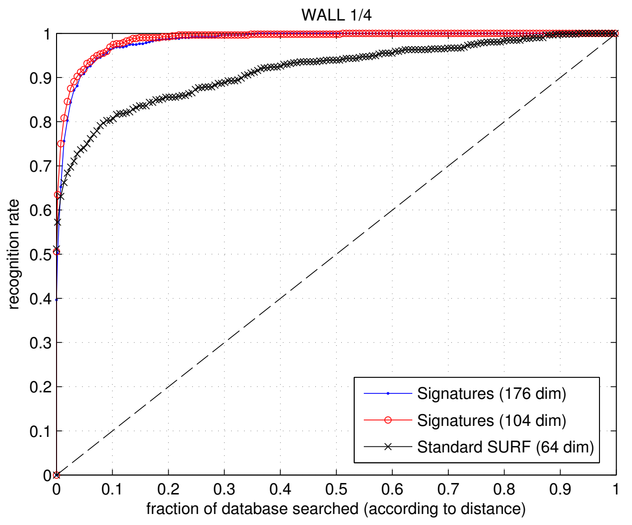


Figure 9. ROC curve comparing the robustness of SURF and Signatures on the Wall dataset. Both Signatures of dimension 176 and 104 clearly outperform SURF.

References

[1] J. Bader and E. Zitzler. HypE: Fast Hypervolume-Based Multiobjective Search Using Monte Carlo Sampling. TIK Report 286, Institut für Technische Informatik und Kommunikationsnetze, ETH Zürich, Oct. 2006.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. SURF: Speeded Up Robust Features. *Computer Vision and Image Understanding*, 10(3):346–359, 2008.

[3] M. Calonder, V. Lepetit, and P. Fua. Keypoint signatures for fast learning and recognition. In *European Conference on Computer Vision*, Marseille, France, October 2008.

[4] M. Calonder, V. Lepetit, K. Konolige, J. Bowman, P. Mihelich, and P. Fua. Compact signatures for high-speed interest point description and matching. In *International Conference on Computer Vision*, Kyoto, Japan, September 2009.

[5] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[6] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *Parallel Problem Solving from Nature (PPSN VI)*. Springer, 2000.

[7] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[8] G. Hua, M. Brown, and S. Winder. Discriminant embedding for local image descriptors. In *International Conference on Computer Vision*, 2007.

[9] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 20(2):91–110, 2004.

[10] Z. Michalewicz and D. B. Fogel. *How to Solve It: Modern Heuristics*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2004.

[11] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast Keypoint Recognition using Random Ferns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008. Accepted for Publication.

[12] M. Ozuysal, P. Fua, and V. Lepetit. Fast Keypoint Recognition in Ten Lines of Code. In *Conference on Computer Vision and Pattern Recognition*, Minneapolis, MI, June 2007.

[13] D. Wagner, T. Langlotz, and D. Schmalstieg. Robust and Unobtrusive Marker Tracking on Mobile Phones. In *International Symposium on Mixed and Augmented Reality*, Nara, Japan, Sept. 2007.

[14] S. Winder and M. Brown. Learning Local Image Descriptors. In *Conference on Computer Vision and Pattern Recognition*, Minneapolis, MI, June 2007.

[15] E. Zitzler and S. Künzli. Indicator-based selection in multiobjective search. In *Parallel Problem Solving from Nature (PPSN VIII)*. Springer-Verlag, 2004.

[16] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. pages 95–100, Barcelona, Spain, 2002. Center for Numerical Method (CIMNE).