Pattern Recognition Letters 106 (2018) 53–60

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Efficient adaptive non-maximal suppression algorithms for homogeneous spatial keypoint distribution

Oleksandr Bailo, Francois Rameau*, Kyungdon Joo, Jinsun Park, Oleksandr Bogdan, In So Kweon

School of Electrical Engineering, KAIST, Daejeon 34141, Republic of Korea

ARTICLE INFO

Article history:
Received 10 May 2017
Available online 26 February 2018

MSC:
41A05
41A10
65D05
65D17

Keywords:
Adaptive non-maximal suppression
Point detection
SLAM

ABSTRACT

Keypoint detection usually results in a large number of keypoints which are mostly clustered, redundant, and noisy. These keypoints often require special processing like Adaptive Non-Maximal Suppression (ANMS) to retain the most relevant ones. In this paper, we present three new efficient ANMS approaches which ensure a fast and homogeneous repartition of the keypoints in the image. For this purpose, a square approximation of the search range to suppress irrelevant points is proposed to reduce the computational complexity of the ANMS. To further speed up the proposed approaches, we also introduce a novel strategy to initialize the search range based on image dimension which leads to a faster convergence. An exhaustive survey and comparisons with already existing methods are provided to highlight the effectiveness and scalability of our methods and the initialization strategy.

1. Introduction

Keypoint detection is often the first step for various tasks such as SLAM [14], panorama stitching [4], camera calibration [3], and visual tracking [5,12]. Therefore, this stage potentially affects the robustness, stability, and accuracy of the aforementioned applications. In the past decade, we have witnessed significant advances in keypoint detectors leading to major improvements in terms of accuracy, speed, and repeatability. But while the detection of keypoints has been intensively studied, ensuring their homogeneous spatial distribution has attracted a rather low level of attention. It is well known that spatial point distribution is crucial to avoiding problematic cases like degenerated configurations (for structure from motion or SLAM) or redundant information (i.e. cluster of points) as depicted in Fig. 1. Moreover, a homogeneous and unclustered point distribution might speed up most computer vision pipelines since a lower number of keypoints is needed to cover the whole image. One of the most effective solutions to ensure well-distributed keypoint detection is to apply an Adaptive Non-Maximal Suppression (ANMS) algorithm on the keypoints extracted by a detector. However, despite all the advantages offered by such approaches, these methods have been rarely used in practice due to their high computational complexity. To overcome this limitation we propose three novel approaches called Range Tree ANMS (RT ANMS), K-d tree ANMS (K-dT ANMS), and Suppression via Square Covering (SSC). The developed algorithms aim to efficiently select the strongest and well-distributed keypoints across the image. We achieve such performance using a square search range approximation which is initialized in an optimal and intuitive manner (see Fig. 2).

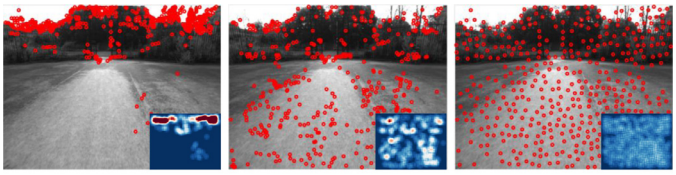
An abundant number of experiments are used to demonstrate the relevance of our ANMS algorithms in terms of speed, spatial distribution, and memory efficiency. Furthermore, we experimentally highlight that ANMS is a beneficial step for SLAM, which drastically improves the accuracy of the motion estimation while using a restricted number of keypoints. To sum up, the contributions of this paper are the following:

- Three novel and efficient ANMS algorithms
- A new and optimal initialization of the search range
- An extensive series of experiments against state-of-the-art
- Efficient and optimized ANMS codes are made available at https://github.com/BAILOOL/ANMS-Codes.

This paper is organized as follows. In Section 2, we provide an extensive literature review of existing approaches. The notations as well as proposed methods are introduced in Section 3. Finally, a large number of experiments is provided in Section 4 followed by a brief conclusion (Section 5).

* Corresponding author.
E-mail address: frameau@kaist.ac.kr (F. Rameau).

https://doi.org/10.1016/j.patrec.2018.02.020
0167-8655/



(a) (b) (c)

Fig. 1. Keypoint detection: (a) TopM NMS, (b) bucketing, (c) proposed ANMS. The bottom right subimage represents the coverage and clusteredness of keypoints computed using a Gaussian kernel. The red color in the subimage stands for a dense cluster of points, while the blue color represents an uncovered area. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

2. Related work

In this section, we report existing methods that have been developed to improve the spatial distribution of keypoints. These approaches can be divided into three categories: bucketing approaches, Non-Maximal Suppression (NMS), and ANMS.

2.1. Bucketing approach

Currently, the bucketing-based point detection approach [10] is the most common technique used to ensure good repartition of the keypoints. This approach is relatively simple: the source image is partitioned into a grid and keypoints are detected in each grid cell. The bucketing-based approach is efficient for detecting keypoints all over the image, however, it is unable to avoid the presence of redundant information (i.e. clusters of keypoints).

2.2. Non-maximal suppression

NMS (also referred to as TopM) is often used to remove a large number of keypoints which are mostly redundant or noisy responses of the keypoint detectors. The most common approach for NMS [15] consists of suppressing the weakest keypoints using an empirically determined threshold. Thereafter, the clusteredness is often reduced by suppressing the keypoints which do not belong to a local maximum in a particular radius. NMS is a straightforward and fast way to reject unnecessary corners, but, in many real case situations, this approach leads to a very limited spatial dissemination of the keypoints (see Fig. 1(b)).

It should be noted that certain works have recently attempted to improve the NMS stage by introducing a novel adaptive cornerness score calculation taking into consideration the local contrast around the keypoints [16]. Thus, these approaches tend to improve the spatial distribution as well as the robustness against illumination variations. However, they suffer from the point clustering effect inherent to NMS approaches.

2.3. Adaptive non-maximal suppression

ANMS methods have been developed to tackle the aforementioned drawbacks. These techniques enforce better keypoint spatial distribution by jointly taking into account the cornerness strength and the spatial localization of the keypoints. The very first ANMS approach was proposed by Brown et al. [4]. The authors initially introduced this concept to robustify the image matching for panorama stitching. In that work, the keypoints are suppressed based on their corner strength and the location of the closest strong keypoint. Unfortunately, the original implementation of this ANMS has a quadratic complexity which is not suitable for real-time applications such as SLAM.

To overcome this problem, multiple attempts to reduce the computational time of ANMS have been investigated. For instance, Cheng et al. [7] proposed an algorithm using a 2-dimensional k-d Tree for space-partitioning of high-dimensional data. Using this data structure, the keypoints are separated into rectangular image regions. Then, from each cell, the strongest features are selected as the output sample set. This algorithm was extended by Behrens et al. [1] using a general tree data structure. While these methods perform faster than the traditional ANMS [4], they do not necessarily output homogeneously distributed points.

More recently, Gauglitz et al. [8] have proposed two complementary approaches that reportedly perform in a subquadratic run time. In the first approach, the authors have chosen to use an approximate nearest neighbor algorithm [6] which relies on a randomized search tree [17]. The second algorithm named Suppression via Disk Covering (SDC) aims to further boost the performance of the ANMS. The algorithm simulates an approximate radius-nearest neighbor query by superimposing a grid onto the keypoints and approximating the Euclidean distance between keypoints by the distance between the centers of the cells into which they fall.

Our proposed approaches tackle the limitations of previous works while maintaining favorable efficiency and scalability.

3. Methodology

In this section, we describe a problem statement and propose several efficient algorithms which ensure a homogeneous repartition of keypoints in the image. Specifically, we cover ANMS based on Tree Data Structure (TDS) (includes K-dT and RT ANMSs) followed by Suppression via Square Covering (SSC). Lastly, we

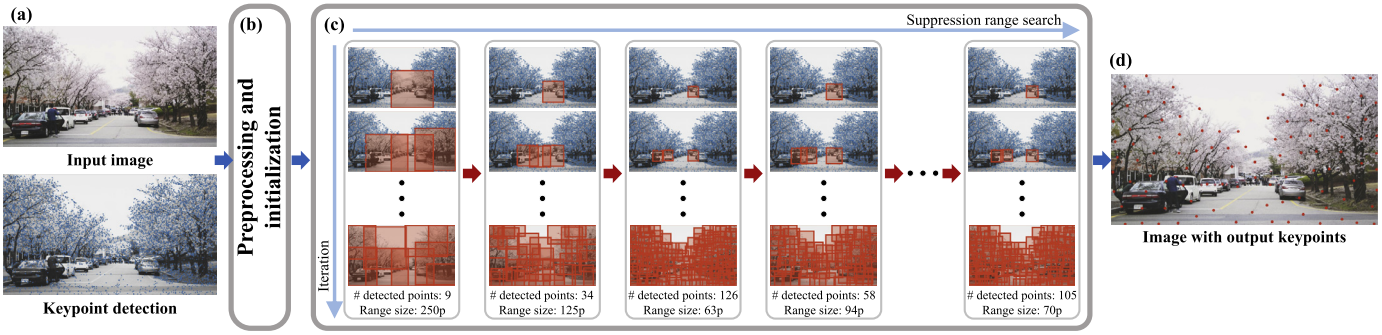


Fig. 2. Algorithm's workflow: (a) keypoint detection in the original image (depicted in blue), (b) sorting keypoints by strength and initialization of the search range, (c) conceptual representation of our ANMS algorithm where: every column represents the search range guess (orange boxes) through a binary search process iterated until queried number of points is reached (100 in this example); while every row depicts the iterations through input points, (d) final result where the red dots represent the selected keypoints. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 1
TDS time and storage analysis.

	K-d Tree		Range Tree	
	Time	Storage	Time	Storage
Insert	$O(\log n)$	$O(n)$	$O(\log^d n)$	$O(n \log^{d-1} n)$
Query	$O(n^{1-1/d} + card(\mathbf{P}_w))$		$O(\log^d n + card(\mathbf{P}_w))$	
Delete	$O(\log n)$		$O(\log^d n)$	

provide a derivation of the initialization of the search range to further speed-up proposed algorithms.

3.1. Problem statement

Most of the recent ANMS approaches share a common pipeline. The set of two-dimensional (d=2) input keypoints $\mathbf{P}_{in} = \{p^i_{in}\}^n_{i=1}$ of size $n = card(\mathbf{P}_{in})$ (where $card(.)$ stands for cardinality operator) is extracted by the detector — and sorted according to the cornerness score of the points. Further, the keypoints in $\mathbf{P}_{in}$ are iteratively processed to compute a smaller and better-distributed set of output keypoints $\mathbf{P}_{out} = \{p^i_{out}\}^m_{i=1}$ of size $m = card(\mathbf{P}_{out})$, where m is defined by the user. The output set of points ensures a good spatial coverage all over the image while avoiding clustering. This homogeneous point distribution is enforced by a spatial consistency check in an adaptive search range of size w (w is the radius of a circle or half the side of a square depending upon what approach is used) defining the suppression neighborhood around a candidate point p_{in}. The radius w is adjusted until the number of retrieved points is close to m according to a certain threshold $m \pm \epsilon_t$, where ϵ_t represents user-defined tolerance threshold.

3.2. ANMS based on tree data structure

Using a data structure is a common way to approach the ANMS problem [8]. However, previous attempts have resulted in relatively inefficient implementations (Section 2). In addition, as observed in [1], after the ANMS step, there are still regions in the image containing a high level of clusteredness. In this section, we propose an efficient algorithm which relies on more suitable data structures and maintains good spatial keypoint distribution. K-dimensional Tree [13] (K-dT) and Range Tree [2] (RT) have been used for this purpose.

First, K-dT is a binary search tree where the data in each node is a K-dimensional point in space. Using this data structure allows space partitioning to organize points in a K-dimensional space. This partitioning can be used to efficiently retrieve the set of points $\mathbf{P}_w$ which falls into a defined range around a particular point. On the other hand, RT is an alternative to K-dT. RT is a binary search tree where the data in each node contains an associated structure that is a $(d-1)$-dimensional RT. Compared to K-dTs, RTs offer faster query times in exchange for worse storage complexity (see Table 1). While these two data structures are essentially different, from the high-level perspective, the algorithm is generic and appropriate for any data structure that is capable of retrieving the set of points within the defined range. Therefore, we describe both proposed algorithms (i.e. K-dT ANMS and RT ANMS) within this subsection.

The TDS is built on keypoints $\mathbf{P}_{in}$ sorted in decreasing order of strength (i.e., cornerness score). This TDS is used in our algorithm as a way to efficiently obtain the nearest neighbors of a particular keypoint given a search range. This search range is determined by the binary search that tries to guess the most appropriate search range w to satisfy the queried number of keypoints. For every w guess, the nearest neighbors of each keypoint (processed in a decreasing order of strength) are suppressed in a way that they will not be considered in further iterations under the selected w. For this purpose, the index list $\mathbf{Idx}_s$ is used to keep track of the uncovered keypoints. The binary search terminates when the number of retrieved keypoints is close to the number of queried keypoints m according to a tolerance threshold $m \pm \epsilon_t$. The outline is provided in Algorithm 1.

Algorithm 1: ANMS based on TDS.

Input: keypoints $\mathbf{P}_{in}$ extracted by the detector
Output: spatially distributed keypoints $\mathbf{P}_{out}$
sort $\mathbf{P}_{in}$ by strength
build TDS on sorted $\mathbf{P}_{in}$
initialize binary search boundaries (Sec. 3.4)
while *binary search for search range* w **do**
  $\mathbf{P}_{out} = \emptyset$
  initialize $\mathbf{Idx}_s$ with all as selected
  for $p_i \in \mathbf{P}_{in}$ **do**
    if $p_i \in \mathbf{Idx}_s$ **then**
      $\mathbf{P}_{out} = \mathbf{P}_{out} \cup p_i$
      $\mathbf{P}_w = TDS.\text{query}(p_i, w)$
      $\mathbf{Idx}_s = \mathbf{Idx}_s \setminus \mathbf{P}_w$
  if $|card(\mathbf{P}_{out}) - m| \leq \epsilon_t$ **then return** $\mathbf{P}_{out}$

The proposed algorithm has similarities to the algorithm presented in [8] where the authors have chosen to use an approximate nearest neighbor algorithm which relies on a randomized search tree. However, that algorithm [8] is not optimally efficient since it performs both query and delete operations for each candidate keypoint in $\mathbf{P}_{in}$ per radius guess. Furthermore, it requires dynamically adding/removing keypoints to the tree which drastically slows performance. In contrast, our algorithms achieve comparable results with a single query operation per search range guess, which makes it more efficient and scalable.

3.3. Suppression via Square Covering

We have compared both K-dT ANMS and RT ANMS and observed similar performance in terms of keypoint repeatability and clusteredness (see Section 4.4). It is worth mentioning that while in the case of K-dT the range of search is defined by the radius around the candidate point, RT uses a square approximation of the search range. This square approximation can potentially boost the speed performance of the ANMS.

One of the key approximations which makes SDC [8] efficient is a radius-nearest neighbor query, by superimposing a grid G_w onto the keypoints and approximating the Euclidean distance between keypoints by the distance between the centers of the cells into which they fall. While this approximation performs well, it still requires computing the Euclidean distance between a large number of keypoints. Therefore, it is a crucial concern since the number of computations increases as the number of keypoints grows.

To tackle the aforementioned problem, we propose to apply the square approximation for the SDC [8] algorithm. In particular, once the grid G_w is set, we try to cover the cells which lie within $2w$ (determined by binary search) regardless of where exactly the

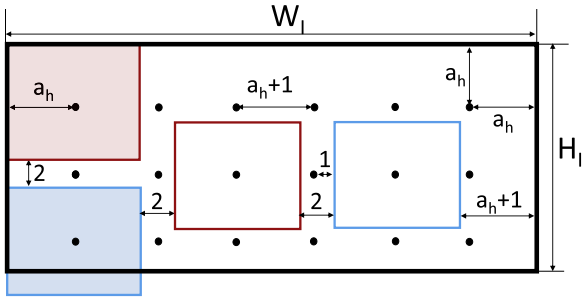


Fig. 3. Graphical representation of the optimal point distribution. Bounding boxes of different colors represent the search range around the candidate points.

points are located inside this square range of coverage. This drastically boosts the performance of the algorithm since covering of the cells is simply performed by traversing through a square search range without the need for Euclidean distance computation. The pseudo-code is in Algorithm 2.

Algorithm 2: Suppression via Square Covering(SSC).

Input: keypoints $\mathbf{P}_{in}$ extracted by the detector
Output: spatially distributed keypoints $\mathbf{P}_{out}$
sort $\mathbf{P}_{in}$ by strength
initialize binary search boundaries (Sec. 3.4)
while *binary search for suppression side* w **do**
  set resolution of grid $G_w = w/2$
  uncover cells of G_w
  $\mathbf{P}_{out} = \emptyset$
  for $p_i \in \mathbf{P}_{in}$ **do**
    if *cell* $G_w[p_i]$ *is* ***not*** *covered* **then**
      $\mathbf{P}_{out} = \mathbf{P}_{out} \cup p_i$
      cover cells of G_w around p_i with square of side $2w$
  if $|card(\mathbf{P}_{out}) - m| \leq \epsilon_t$ **then return** $\mathbf{P}_{out}$

3.4. Initialization of search range

Similar to our proposed algorithms, the SDC [8] uses binary search to guess the appropriate search range. In this previous work, the upper bound a_h of the binary search is set to image width W_I, while the low one a_l is set to 1. This often results in unnecessary iterations and decreases the convergence speed. To tackle this problem, we propose a novel and elegant way to precompute the bounds for the binary search which drastically decreases the number of iterations until convergence and, in turn, improves the speed of the algorithm.

Our problem statement is the following, we want to homogeneously distribute the m queried number of points on the image without any clusters. To do so, we try to cover the image with squares of side $2a_h$ with a minimal distance between the square centers $a_h + 1$ (see Fig. 3). Given $2a_h$ and W_I, we can calculate the maximum number of squares that perfectly fit in a row of the image. We define this row as a set of squares placed at the same height in the image where the first and the last square in a row are perfectly aligned with image borders. If there are q points (i.e. square centers) inside each row, then there are $q-1$ distances in each row between these points. In addition, the left and right extreme points are located at a distance a_h from the left and right borders of the image. Thus, we can express the image width W_I in terms of a_h and q:

$$W_I = 2a_h + (a_h + 1)(q - 1), \tag{1}$$

hence, from Eq. (1) the number of points in each row is:

$$q = \frac{W_I - a_h + 1}{a_h + 1}. \tag{2}$$

Similarly, the maximum number of square centers l possibly fitting within the image height H_I is:

$$H_I = 2a_h + (a_h + 1)(l - 1). \tag{3}$$

The queried number of points m is equal to the product of q and l. By substituting $l = \frac{m}{q}$ to Eq. (3) and substituting q from Eq. (2), we obtain the following equation:

$$(m-1)a_h^2 + a_h(W_I + 2m + H_I) + m + W_I - H_I W_I = 0. \tag{4}$$

Solving this equation for a_h yields two solutions, one of which is always negative, while the other one gives us the final estimation of the square side:

$$a_h = -\frac{H_I + W_I + 2m - \sqrt{\Delta}}{2(m-1)}, \tag{5}$$

where the discriminant of the quadratic Eq. (4) is:

$$\Delta = 4W_I + 4m + 4H_I m + H_I^2 + W_I^2 - 2W_I H_I + 4W_I H_I m. \tag{6}$$

It is worth mentioning, while the solution tries to allocate as many points as possible, it does not guarantee that the number of points on the image will be exactly equal to m. This happens for several reasons. First of all, the fraction $\frac{m}{q}$ might not produce an integer value for l. Secondly, during the code implementation, we round the obtained value for a_h since the minimum unit of an image is 1 pixel.

The lower bound a_l of the binary search can be determined by looking closely at the worst possible point distribution. This happens when all n input points are located in a single square on the image with no space between them. Given such distribution, we want to retrieve at least m queried points by filling this space with the smallest possible squares of side $2a_l$. This can be mathematically expressed as $m(2a_l)^2 = n$. Therefore, the equation for the lower bound of the binary search is the following:

$$a_l = \frac{1}{2}\sqrt{\frac{n}{m}}. \tag{7}$$

4. Results

4.1. Time and storage complexity

The detailed time complexity analysis is provided in Table 2. All of the presented algorithms (listed in 'Method' column) rely on the preprocessing (i.e. sorting by strength) input keypoints. For this purpose, we utilize a sorting algorithm with an average performance of $O(n\log n)$. Additionally, K-dT and RT ANMSs rely on TDS which has to be populated with the input keypoints. This is performed by inserting (see Table 1 for complexity) n number of keypoints one by one into a data structure resulting in overall $O(n\log n)$ and $O(n\log^d n)$ complexity, respectively. The query time for each algorithm to select appropriate keypoints is stated in the 'Query' column in Table 2. Specifically, the TopM algorithm simply retrieves m number of keypoints from an already sorted list in $O(m)$. The traditional ANMS [4] (designated 'Brown' in our experiments) algorithm requires the computation of the minimum distance between every keypoint which requires $O(n^2)$ followed by sorting in $O(n\log n)$ and keypoint retrieval in $O(m)$. Since the rest of the algorithms (SDC, K-dT ANSM, RT ANMS, and SSC) rely on a binary search algorithm to find the appropriate search range in $O(\log w_{ini})$, the total query time complexity can be obtained by multiplying the number of search range guesses with the complexity of keypoint selection per every guess. The total time complexity

Table 2
Time and storage complexity.

Method	Time complexity				Storage complexity
	Preprocess	Build	Query	Total (approximated)	
TopM	$O(n\log n)$	–	$O(m)$	$O(n\log n)$	$O(n+m)$
Brown	$O(n\log n)$	–	$O(n^2+n\log n+m)$	$O(n^2)$	$O(n+m)$
SDC	$O(n\log n)$	–	$O(\log w_{ini}\cdot(n+m/\varepsilon_r))$	$O(n\log n+\log w_{ini}\cdot(n+m/\varepsilon_r))$	$O(n+m)$
K-dT ANMS	$O(n\log n)$	$O(n\log n)$	$O(\log w_{ini}\cdot(n+n^{1-1/d}+\sum card(\mathbf{P}_w)))$	$O(n\log n+\log w_{ini}\cdot(n+\sum card(\mathbf{P}_w)))$	$O(n+card(\mathbf{P}_w)+m)$
RT ANMS	$O(n\log n)$	$O(n\log^d n)$	$O(\log w_{ini}\cdot(n+\log^d n+\sum card(\mathbf{P}_w)))$	$O(n\log^d n+\log w_{ini}\cdot(n+\sum card(\mathbf{P}_w)))$	$O(n\log^{d-1} n+card(\mathbf{P}_w)+m)$
SSC	$O(n\log n)$	–	$O(\log w_{ini}\cdot(n+4m))$	$O(n\log n+\log w_{ini}\cdot(n+4m))$	$O(n+m)$

listed in 'Total (approximated)' gives us the following insight into the algorithm's performance. Obviously, the TopM approach clearly outperforms all other methods in terms of speed due to its simplicity. Furthermore, SDC, K-dT ANMS, RT ANMS, and SSC are certainly asymptotically faster than traditional 'Brown' [4].

The storage complexity evaluation is shown in Table 2. Methods that do not rely on any data structure (e.g., TopM, 'Brown' [4], SDC, SSC) at most occupy memory necessary to incorporate a number of input and output keypoints resulting in $O(n+m)$ complexity. These methods surely demonstrate better storage complexity compared to K-dT and RT ANMSs which additionally require memory for storing TDS (see Table 1).

Overall, due to the sophisticated time complexity estimation, it is challenging to highlight a clear winner among the fastest ANMS algorithms (SDC, K-dT ANMS, RT ANMS, and SSC). In order to provide a qualitative evaluation of the algorithms, we have performed an extensive evaluation of all methods.

4.2. Synthetic and real experiments

First of all, to fairly assess the speed performance of the different algorithms, a large series of synthetic experiments has been performed. For this purpose, a set of randomly distributed 2D points is generated on a synthetic image of resolution $1280\times720p$. Further, a random cornerness score is individually assigned to every point to simulate the behavior of keypoint detection in a natural image. The number of 2D points is in range $[800, 11000]$ with a step of 100. Every test is repeated 1000 times to ensure an unbiased estimation of the algorithms' speed. The queried number of points is fixed to 800, while the search range w is initialized to image width. We intentionally did not use our initialization technique (Section 3.4) for this test to provide a fair comparison with SDC [8]. It should be noted that Brown [4] has been removed from these experiments for the sake of clarity (i.e., scale inconsistency) since this method is significantly slower than the proposed approaches.

The mean computational time and the standard deviation against the number of points per iteration are available in Fig. 4(a) and (b) respectively. Through this experiment, it is noticeable that the TopM algorithm drastically outperforms more sophisticated approaches (but provides a very unsatisfying point distribution in practice, see Section 4.4). Other algorithms show interesting characteristics. SSC is indisputably more efficient than any existing algorithms both in stability (i.e., the standard deviation remains very low) and speed. On the other hand, SDC demonstrates satisfying results but suffers from a lack of stability for a low number of input keypoints. Indeed, this approach is more efficient than RT ANMS when the number of input points is large, however, this tendency is reversed when the detected keypoints do not exceed 5000. Moreover, SDC is less scalable than our SSC approach. Finally, despite its efficiency for a small number of points, K-dT ANMS loses its advantage for more than 2000 points.

While the assessment with synthetic data is a good evaluation showing clear tendencies, the distribution of keypoints in real images can be different than the ones obtained synthetically. Therefore, we propose an extensive series of evaluations using real images. For this purpose, we select 1000 images from KITTI [9] (Sequence 00), and detect keypoints using FAST [15] with a threshold $t_h = 5$. Such relatively low threshold results in a large number of detected keypoints (i.e., >10000 keypoints per image). Subsequently, the keypoints are sorted by their strength . Further, we iteratively select a fixed number of the strongest keypoints starting from 100 until reaching 10000. The step of such selection is 100, which results in 100 tests per image. The ANMS algorithms return a fixed percentage of the input number of keypoints. For instance, for 1000 input keypoints and a queried percentage of 10%, the number of queried keypoints is 100. We have applied 10 different ratios in range [10%, 100%] with a step of 10%. Several representative results from these evaluations are provided in Fig. 5.

This extensive evaluation demonstrates that SSC clearly outperforms all other methods in terms of speed. Overall, different conclusions from those obtained with the synthetic experiments can be drawn. Indeed, SDC remains efficient for a relatively small number of queried keypoints (Fig. 5(a)) but tends to be less effective when a large number of input and output keypoints are processed. This can be explained by the substantial number of Euclidean distances comparison to be computed for a dense set of input keypoints (Fig. 5(b) and (c)). In this respect, our RT ANMS scales more efficiently even when many outputs points are requested (see Fig. 5(b) and (c)). Finally, our K-dT ANMS becomes inefficient for a large number of points due to the relatively slow query time of this data structure (see Fig. 5(c)).

4.3. Effect of proposed initialization

In this experiment, we evaluate the impact of our initialization on the speed of the different methods. For this purpose, the same real-image experimental setup described in Section 4.2 is used. However, in this case, we employ our initialization approach (Section 3.4). Two criteria have been utilized to determine the advantages offered by this technique. The first is the number of iterations needed to reach the number of queried points (see Fig. 6). The second one is the overall speed-up provided to each method (see Table 3).

For every single method, the number of necessary iterations has been reduced by a factor of three, leading to a significant speed-up. It is noticeable that certain approaches are more affected by this initialization. For instance, this is the case for the K-dT ANMS approach which has been sped-up by a factor of $2.6\times$. However, some other algorithms such as our RT ANMS has been moderately improved by our bounds calculation. This can be explained by the nature of the RT structure itself. In fact, with a closer initialization (i.e., a smaller search range), the total number of expensive queries increases.

4.4. Clusteredness

The main advantage of using an ANMS strategy is the unclustered and well-distributed set of keypoints resulting from this

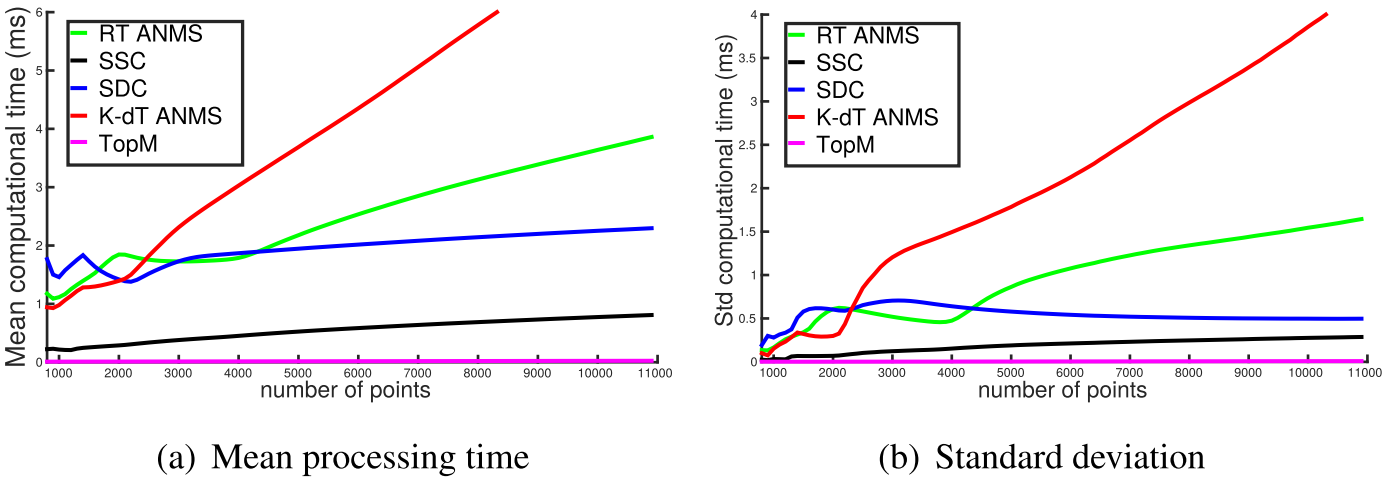


Fig. 4. Comparison of methods on synthetic data: (a) mean processing time, (b) standard deviation.

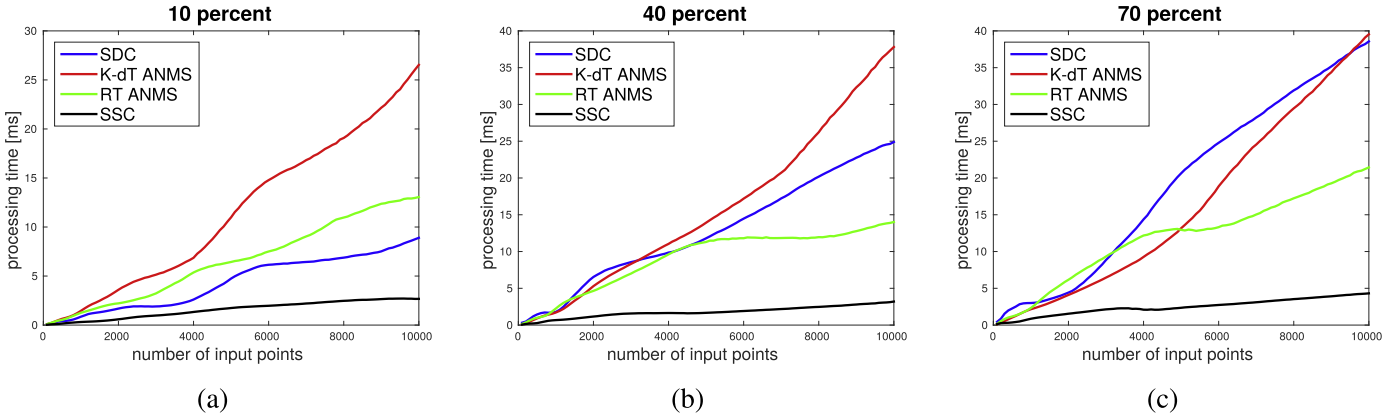


Fig. 5. Mean processing time *vs.* number of input keypoints for 1000 images. Subfigures (a)-(c) show linear scale of the *y* axis.

Table 3
Speedup provided by initialization (average over 1000 images).

Method	Without initialization (ms)	With initialization (ms)	Speedup
SDC	7.4	3.1	2.4×
K-dT ANMS	17.5	6.8	2.6×
RT ANMS	8.9	7.3	1.2×
SSC	2.0	1.4	1.4×

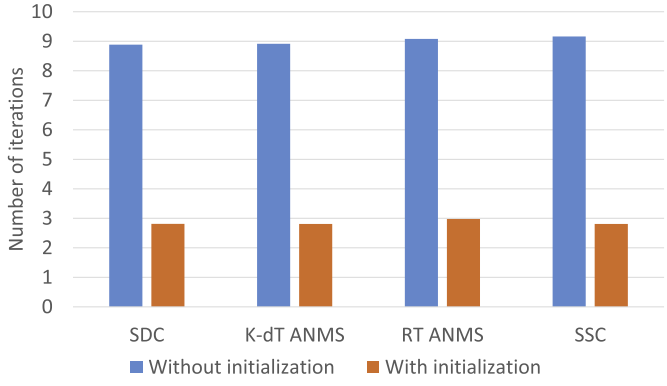


Fig. 6. Number of iterations until convergence (with and without initialization).

process. Indeed, this feature allows us to avoid redundant information typically occurring with commonly used approaches such as bucketing keypoint detection and standard NMS. To evaluate the clusteredness we have reproduced the experiment suggested in [16], where the authors propose an appropriate metric to evaluate this criterion. For this evaluation, the image is divided into a regular grid of 10×10 cells to compute the number of points lying in every single cell. The standard deviation of the number of corners per cell is utilized as the clusteredness metric since it is representative of the homogeneity of the spatial distribution.

To provide a statically valid evaluation of the clusteredness for every single approach, 2000 randomly selected images from the KITTI dataset [9] are used. In this experiment, $t_h = 12$ and the number of queried keypoints m varies between 100 to 700. The obtained results are visible in Fig. 8. We can clearly notice that all the ANMS approaches provide similar outputs in terms of spatial distribution which can also be observed in Fig. 7. As to the bucketing approach (grid size is 7×5), it produces better spatial distribution than TopM, but cannot meet the performance of the ANMS strategies. This can be explained by the fact that bucketing approach is designed to ensure good spatial distribution, but does not solve the problem of point clusters.

(a) (b) (c)

Fig. 7. Keypoint detection: (a) K-dT ANMS, (b) RT ANMS, (c) SSC. The red dots represent selected keypoints. In this experiment, $t_h = 12$ and $m = 100$.

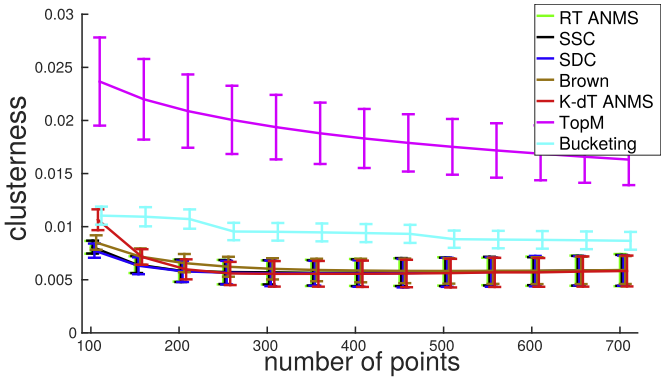


Fig. 8. Mean and standard deviation of the clusteredness over 1000 images.

4.5. Application to SLAM

SLAM is one of the applications where the spatial distribution of the keypoints on the image is crucial. Therefore, we have included our ANMS solutions in a stereo-SLAM algorithm which is conceptually close to S-PTAM [14]. Specifically, the keypoints are detected on both stereo-images using the FAST ($t_h = 12$) and filtered by our ANMS algorithms to reach 750 points. These stereo points are matched together using a line search strategy and triangulated to initialize the 3D map. Motion tracking is performed using a RANSAC P3P [11] algorithm. Finally, the mapping is achieved by refining the structure and the motion together via a local bundle adjustment scheme. For this evaluation, we have utilized all the training sequences from the KITTI dataset [9] where an accurate ground truth is provided. The mean translation (in percentage) and rotation (in degree) error per sequence are computed with the metrics recommended by [9].

The results of the entire experiment are available in Fig. 9. Regarding the translational error, a clear tendency is noticeable. For instance, the TopM algorithm is particularly inefficient in light of the other approaches. On the other hand, the bucketing approach tends to perform better than the TopM approach but never provides better results than the ANMS methods. All the ANMS approaches provide comparable results. The same tendency is observed for the rotation estimation. However, for rotation, the detection of well-distributed keypoints is less crucial. The error discrepancy between the different sequences may be justified by the various contexts in which the sequences have been acquired. For instance, in Seq00 the large rotational error can be explained by the high quantity of turns in the sequence, while Seq04 (which admits a very low rotational error) mostly consists of a short and straight line. Moreover, Seq01 is interesting to analyze because it is probably the most challenging - the vehicle is going at high speed through a relatively empty scene (low texture). These factors make the keypoints particularly difficult to track. Under these conditions, ANMS algorithms show even more significant improvements. Another observation is the improved robustness to moving objects. Our approaches also show very promising results in sequences containing one or more moving objects (for instance in Seq04). With ANMS only a few keypoints are detected on moving objects, while their majority belong to the rigid background, therefore, the outliers are more efficiently removed by a robust estimation step (RANSAC in our SLAM). Finally, in Fig. 9, the slight error discrepancy between ANMS methods is mostly due to the inherent randomness of the point tracking strategy, noise, and numerical error typical of real image experiments. Nevertheless, we can certainly conclude that all the ANMS approaches - compared in this paper - significantly improve the SLAM algorithm in a very similar manner.

In Fig. 10, we propose a qualitative comparison of our SSC algorithm against the bucketing strategy. For this estimation, we use the New College dataset [18] (see Fig. 1) consisting of 50,000 stereo image pairs, covering 2.5km with a hand-held stereo camera (multiple loops and challenging scenarios). Through this experiment, it is clear that our ANMS approach significantly reduces drift over the sequence compared to the bucketing approach. This drift is particularly obvious in the side view (see Fig. 10(b)). Note that the TopM algorithm is not depicted in this figure for sake of clarity (very large drift).

4.6. Discussion on proposed methods

Certainly, ANMS approaches are beneficial under specific contexts and conditions. It is appropriate for pose estimation (SLAM, panorama stitching, etc.), self-calibration, and Structure from Motion (SfM). Similarly, Schauwecker et al. [16] have demonstrated that a good dissemination of the points in the images resulted in a better sparse stereo matching. However, ANMS is not limited to these topics and can be appropriate for many real-time approaches. For example, it might be the case for Bag-of-Word place recognition, where well-distributed points can lead to a stronger description of the image. While the authors have originally developed SDC [8] for planar tracking purposes, we believe that ANMS might be counter-productive for visual tracking under certain conditions (i.e. small target, cluttered scene). Other techniques requiring a dense cluster of points on a salient part of the image (i.e. point based obstacle detection) would probably not be improved by ANMS.

In this paper, we have proposed three ANMS techniques named K-dT ANMS, RT ANMS, and SSC to homogeneously distribute keypoints on the image. While ANMS methods provide visually and statistically (analyzed by Z-test) similar outputs in terms of spatial distribution, SSC demonstrates the best time and scalability performance. Therefore, this algorithm is advisable for use in case an application requires real-time performance even when the number of input points is relatively high. This would include, for example, real-time SLAM or visual odometry. On the other hand, since K-dT and RT ANMSs are based on TDS to store the input points, they can be used for situations when keypoints need to be reused. A good example is a large scale SfM where many re-projection on the images have to be performed to aggregate new images. Thus, these approaches can be accelerated by using the same structure for keypoints detection and for point matching. Compared to K-dT ANMS, RT ANMS offers faster query time but requires more stor-

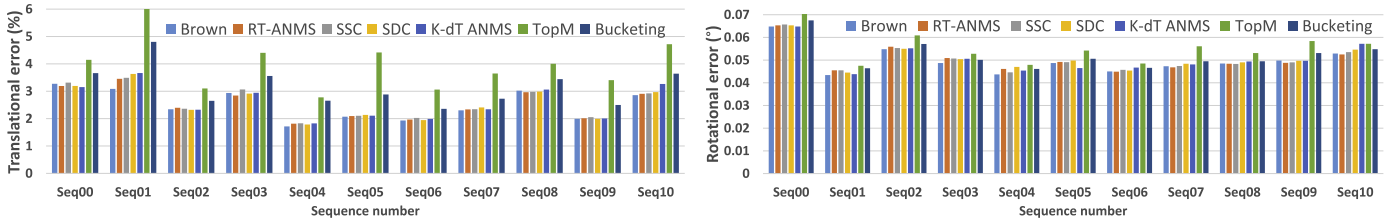


Fig. 9. Experimental results of different methods on SLAM: (a) mean translational error, (b) rotational error.

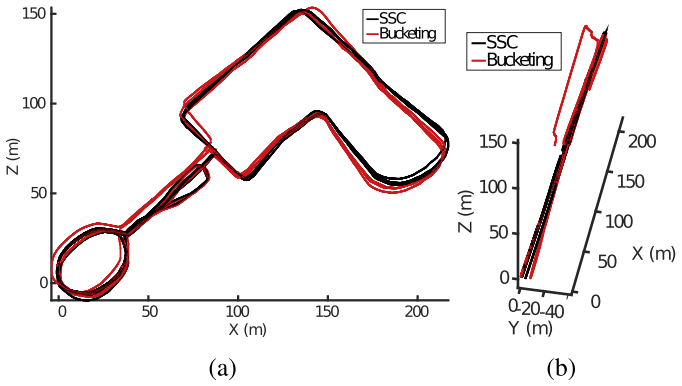


Fig. 10. Trajectories computed on the New College dataset using our SSC approach and the bucketing strategy: (a) top view, (b) side view.

age memory. Therefore, a user should consider this tradeoff when choosing among the proposed ANMS based on TDS.

5. Conclusion

In this paper, we have presented three novel ANMS techniques (codes are provided) to homogeneously distribute detected keypoints in the image. Through an extensive series of experiments, we have highlighted the effectiveness and scalability of our approaches. Furthermore, we have demonstrated the positive impact of our ANMS strategies on visual SLAM. The presented results show that ANMS is a beneficial step for improving SLAM performance. Another major contribution of this paper is the binary search boundaries initialization which drastically reduces the number of iterations needed to retain the queried number of points. The proposed initialization is designed to be suitable for any ANMS relying on binary search.

The current ANMS approaches are designed to handle conventional images, but may perform poorly on non-uniform spatial resolution induced by distortion (e.g. fisheye lens, catadioptric system, etc.). Naturally, the extension of this work will focus on this problem by proposing an ANMS applicable to the unified spherical model.

Acknowledgment

This research was supported by the Shared Sensing for Cooperative Cars Project funded by Bosch (China) Investment Ltd. The second author was supported by Korea Research Fellowship (KRF) Program through the NRF funded by the Ministry of Science, ICT and Future Planning (2015H1D3A1066564).

References

[1] A. Behrens, H. Röllinger, Analysis of feature point distributions for fast image mosaicking algorithms, Acta Polytech. 50 (4) (2010).
[2] R. Berinde, Efficient implementations of range trees(2007).
[3] Y. Bok, H. Ha, I. Kweon, Automated checkerboard detection and indexing using circular boundaries, Pattern Recognit. Lett. 71 (2016) 66–72.
[4] M. Brown, R. Szeliski, S. Winder, Multi-image matching using multi-scale oriented patches, CVPR, 2005.
[5] S. Buoncompagni, D. Maio, D. Maltoni, S. Papi, Saliency-based keypoint selection for fast object detection and matching, Pattern Recognit. Lett. 62 (2015) 32–40.
[6] T. Chan, A minimalists implementation of an approximate nearest neighbor algorithm in fixed dimensions, See https://goo.gl/cvDjAs (2006).
[7] Z. Cheng, D. Devarajan, R. Radke, Determining vision graphs for distributed camera networks using feature digests, EURASIP J. Appl. Signal Process. 2007 (1) (2007). 220–220
[8] S. Gauglitz, L. Foschini, M. Turk, T. Höllerer, Efficiently selecting spatially distributed keypoints for visual tracking, ICIP, 2011.
[9] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? The kitti vision benchmark suite, CVPR, 2012.
[10] B. Kitt, A. Geiger, H. Lategahn, Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme, IV, 2010.
[11] L. Kneip, D. Scaramuzza, R. Siegwart, A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation, CVPR, 2011.
[12] Q. Miao, G. Wang, C. Shi, X. Lin, Z. Ruan, A new framework for on-line object tracking based on surf, Pattern Recognit. Lett. 32 (13) (2011) 1564–1571.
[13] M. Muja, D. Lowe, Scalable nearest neighbor algorithms for high dimensional data, IEEE Trans. Pattern Anal. Mach. Intell. 36 (11) (2014) 2227–2240.
[14] T. Pire, T. Fischer, J. Civera, P. De Cristóforis, J. Berlles, Stereo parallel tracking and mapping for robot localization, IROS, 2015.
[15] E. Rosten, T. Drummond, Machine learning for high-speed corner detection, ECCV, 2006.
[16] K. Schauwecker, R. Klette, A. Zell, A new feature detector and stereo matching method for accurate high-performance sparse stereo matching, IROS, 2012.
[17] R. Seidel, C. Aragon, Randomized search trees, Algorithmica 16 (4) (1996) 464–497.
[18] M. Smith, I. Baldwin, W. Churchill, R. Paul, P. Newman, The new college vision and laser data set, Int. J. Rob. Res. 28 (5) (2009) 595–599, doi:10.1177/0278364909103911.