



Shape binary patterns: an efficient local descriptor and keypoint detector for point clouds

Cristina Romero-González¹ · Ismael García-Varea¹ · Jesus Martínez-Gómez¹

Received: 3 March 2021 / Revised: 14 August 2021 / Accepted: 20 September 2021 /
Published online: 13 January 2022
© The Author(s) 2021

Abstract

Many of the research problems in robot vision involve the detection of keypoints, areas with salient information in the input images and the generation of local descriptors, that encode relevant information for such keypoints. Computer vision solutions have recently relied on Deep Learning techniques, which make extensive use of the computational capabilities available. In autonomous robots, these capabilities are usually limited and, consequently, images cannot be processed adequately. For this reason, some robot vision tasks still benefit from a more classic approach based on keypoint detectors and local descriptors. In 2D images, the use of binary representations for visual tasks has shown that, with lower computational requirements, they can obtain a performance comparable to classic real-value techniques. However, these achievements have not been fully translated to 3D images, where research is mainly focused on real-value approaches. Thus, in this paper, we propose a keypoint detector and local descriptor based on 3D binary patterns. The experimentation demonstrates that our proposal is competitive against state-of-the-art techniques, while its processing can be performed more efficiently.

Keywords Shape binary patterns · Point clouds · Local descriptor · Keypoint detector

1 Introduction

Computer vision is especially relevant in robotics, due to the prominent role of visual information in most robot applications. Thus far, there has been extensive research on the use of vision systems for navigation, localization, manipulation, and human-robot interaction, among others [25]. However, robot vision presents specific constraints that are usually

✉ Ismael García-Varea
Ismael.Garcia@uclm.es

Cristina Romero-González
Cristina.RGonzalez@uclm.es

Jesus Martínez-Gómez
Jesus.Martinez@uclm.es

¹ Computer Systems Department, University of Castilla-La Mancha, Campus Universitario s/n, 02071 Albacete, Spain

out of scope in computer vision applications such as fast computation and low resources consumption.

Computer vision applications have traditionally followed a standard pre-processing pipeline: a) segment an input image, b) detect regions of interest, and c) describe those regions. The first step is usually applied to reduce the search space in the input image. The second step is carried out by detecting keypoints, that is, significant points that can be identified from different viewpoints and which represent areas with salient information. Finally, in the third step, the region around those keypoints must be studied to find a suitable representation that depicts that area with sufficient descriptiveness and distinctiveness. This pipeline has been widely used for tasks like object registration, where the final goal is to find a transformation that overlaps different images of the same object. In this case, it is essential to find representative areas in the object and describe them correctly, to then be able to match the common parts that are visible in the different images.

However, computer vision research has lately been dominated by deep learning techniques, and more specifically, Convolutional Neural Networks (CNNs) [20]. Given enough annotated data and computational capabilities, these solutions offer remarkable accuracy. Nonetheless, there are problems for which the traditional approach is more appropriate. For example, robotic applications are usually limited to the hardware available on the robot itself, and a limited amount of annotated data collected during runtime.

Meanwhile, the use of 3D devices (e.g. Microsoft Kinect or Asus Xtion) is now common for robot vision applications [5, 19, 49, 53], as RGB-D images allow the information perceived within the image to be associated with its 3D position. Consequently, the classic pipeline of segmentation, keypoint detection and local descriptor generation has been adapted to include depth information. The Point Cloud Library (PCL) [44] includes the appropriate tools to process this 3D data.

Binary representation has three clear benefits over other state-of-the-art techniques: efficient computation, low memory requirements, and appropriate encoding for subsequent applications, such as matching. However, research in this type of encoding for 3D information has thus far been limited. Consequently, in this paper we propose a binary representation of the 3D data in point clouds to both detect keypoints and generate local descriptors. In essence, we have designed a binary pattern to encode the shape of the local neighborhood of a point, which can be used as a local descriptor or to analyze whether the point is relevant enough to be used as keypoint. We have also compared the performance of both the keypoint detector and the local descriptor against well-known techniques, obtaining competitive results with significantly lower computational requirements.

The rest of this paper is organized as follows. In Section 2, we present an overview of the current proposals. Our approaches for local descriptors generation and keypoint detection are described in Section 3 and Section 4, respectively. We then compare them against state-of-the-art techniques in Section 5. Finally, Section 6 presents the conclusions.

2 Related work

Image understanding relies on the information extracted from a given input image, and this analysis can be performed using the whole image or small patches around regions of interest. These two approaches are usually referred to as global or local techniques, respectively. While global methods attempt to exploit all the image, local ones use exclusively the information contained in a small region around a given point. This area

is known as local neighborhood, and, in 3D images, is defined by a sphere of radius R centered around the point.

Two of the most important tasks in image understanding are keypoint detection and feature generation, which are usually approached using local techniques. These have been widely studied in the computer vision community [27, 46]. Traditionally, they were approached by describing the local neighborhood with real-value descriptors [4, 22]. Subsequently, however, the interest in binary representations increased due to their characteristics: small memory footprint and fast computation, which make them an appropriate option for real-time applications [24].

In [32], Ojala et al. presented the most significant binary descriptor thus far: the Local Binary Patterns (LBP), a simple and efficient pattern to represent the local neighborhood of a point. The same paper describes how the patterns with all the 1s together identify basic forms (e.g. lines or dots), thus better representing the regions of interest. This subset of patterns is known as uniform patterns. Initially, this descriptor was oriented to 2D texture classification [15, 59], but it has also been successfully applied to multiple tasks, including face recognition [1], shape localization [16] or scene classification [55].

Other binary descriptors for RGB images, such as BRIEF [7], ORB [40], BRISK [21] or FREAK [2], have achieved accuracy comparable to state-of-the-art real-value descriptors in tasks like feature matching [28] and image retrieval [8]. However, research with RGB-D images has mainly been focused on real-value features, such as Spin Images (SI) [17], NARF [48], FPFH [42], or SHOT [50]. In general, these descriptors are computationally demanding in both time and space [3].

To date, a limited number of proposals have been made for the generation of 3D binary descriptors. Some approaches compute LBP in the frequency domain [12, 13], and then construct rotation invariant patterns while testing different possible orientations. Other techniques generate a binary pattern from equidistant points in a sphere of radius R [29, 33] or even from triangular mesh manifolds [54]. These proposals are 3D extensions of LBP with a special focus on the grayscale information around the interest point. Their range of application is reduced to problems where texture classification is essential, such as medical image categorization. Finally, in [35] a binarization of the SHOT descriptor was proposed (B-SHOT) to speed up subsequent tasks like feature matching. However, this proposal cannot take advantage of the fast computation of binary descriptors because it first computes the SHOT feature to then discretize its values.

The developments in keypoint detectors have followed a similar path. Traditionally, SIFT [22] and SURF [4] have been the most commonly used methods, despite their high computational requirements. Recently, new methods based on binary descriptors have been proposed to decrease those requirements, while maintaining similar performance. In 2010, the FAST [39] method was proposed. FAST is a keypoint detector based on the corners found in an image, and was designed specifically for a high processing speed. It determines that a point corresponds to a corner if n contiguous pixels in a circle around the point have a brighter intensity than the center point. To achieve its high speed, it only performs the minimum number of comparisons necessary to determine whether a point is a corner or not. Similar approaches based on the same idea are AGAST [23], which increases performance by providing an adaptive and generic accelerated segment test; ORB [40], which adds an orientation component to FAST for its keypoint detection; and BRISK [21] which is an extension aimed to achieve invariance to scale. A further, less related proposal is FREAK [2], which computes a cascade of binary strings by comparing intensities over a retinal sampling pattern.

More specifically, in 3D, keypoint detectors are focused on finding distinctive shapes within an image based on the 3D surface. These detectors include the MeshDoG [56], which is designed for uniformly triangulated meshes, and is invariant to changes in rotation, translation, and scale; the Laplace-Beltrami Scale-Space (LBSS [52]), which pursues multi-scale operators on point clouds that allow detection of interest regions; the KeyPoint Quality (KPQ [47]) or the Salient Points (SP [9]). Other detectors require a scale value to determine the search radius for the local neighborhood of the keypoint. Examples of these detectors are the Intrinsic Shape Signature (ISS [60]), which relies on Eigen Value Decomposition (EVD) of the points in the support radius; the Local Surface Patches (LSP [10]), based on point-wise quality measurements; and the Shape Index (SI [18]), which uses the maximum and minimum principal curvatures at the vertex. However, none of these methods utilizes binary representations.

Due to the outstanding results of CNNs in computer vision, there has recently been increasing interest in Deep Learning techniques and how to apply them to 3D information [14]. Most of these approaches are based on using the coordinates of the 3D points in an image as input data, and designing an architecture that can use this information for different tasks such as classification or semantic segmentation [36]. For example, inspired by this proposal, PPFNet [11] trains a local network using a set of neighboring points, their normals and point pair features (PPFs [42]). This approach seems to improve previous proposals in recall, but has similar performance to non-Deep Learning approaches in precision, probably due to the limited availability of large 3D datasets.

Current 3D registration methods based on correspondences mix Deep Learning with traditional approaches [58]. Additionally, robotic applications have limited resources to process input data, that is, tasks should be performed in the most efficient way (for example, limiting the use of the GPU only to complex problems). Additionally, in this scenario, solutions must be applied to real-world objects, i.e. there is a limited amount of data available. Consequently, there is still interest in developing efficient hand-crafted local representations.

In this paper, we tackle the challenge of defining an efficient local descriptor and a repeatable keypoint detector. The former is addressed by generalizing the Shape Binary Patterns (SBP) that were first proposed in [37]. The original design of the SBP descriptor was oriented to segmented objects. That approach, however, might not be optimal when dealing with scenes that contain several objects. In these circumstances, clutter hinders the calculation of the density of points for each object and decreases the efficacy of our original proposal. Consequently, in this paper, we generalize our previous proposal to adapt its use to this type of image. We also evaluate its performance with a different benchmark to validate the usefulness of the SBP descriptor. The latter relies on the definition of uniform patterns presented in [38], but here we present a highly revised and more efficient approach.

3 Shape binary patterns

The work presented in this paper is mainly inspired by Local Binary Patterns (LBP) [32], where a pattern T is generated around a given point p_c by computing the difference in the gray value of p_c and a set of N equidistant points on a circle of radius R around that point. The elements with positive differences are represented by 1s and negative ones are encoded as 0s:

$$T = \langle s(g_1 - g_c), \dots, s(g_N - g_c) \rangle \tag{1}$$

where g_i represents the gray level of p_i and

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{2}$$

In point clouds, we could generate a similar descriptor by selecting N equidistant points to the given point p_c using a R -radius sphere, and computing the grayscale difference between each point in $\{p_1, \dots, p_N\}$ and the center p_c . The underlying problem in this approach is the need to find the N points in a specific position on the sphere of radius R . In general, the inherent occlusions of RGB-D images (and 3D models) would render this task impossible. To overcome this problem, in [37] we proposed the Shape Binary Pattern (SBP), a binary descriptor that encodes the presence of points in the local neighborhood of p_c . That version included an automatic radius calculation for the local neighborhood of p_c that worked for the specific task of object registration, when the object had been previously segmented. However, this approach is not recommended when the object is part of a larger scene, as the dimensions of the whole image skew the calculated radius. Consequently, here we aim to present a more general approach with that step removed. Radius R for the local neighborhood is now the only requested input parameter.

The SBP is based on overlapping a 3D grid over the local neighborhood of a given point p_c , and then assigning a binary value to each bin depending on the presence of points inside the bin. The final pattern is constructed by concatenating the values for all the bins. In order to achieve rotation invariance, we need to provide a repeatable local Reference Frame (RF) based just on the local neighborhood of p_c [50].

First, given a set of N points $\mathcal{P}_c = \{p_1, \dots, p_N\}$ corresponding to the nearest points to p_c in a search radius R , the neighbors' covariance matrix M for p_c is computed as:

$$M = \frac{1}{N} \sum_{i=1}^N (p_i - \mu)(p_i - \mu)^T \tag{3}$$

where N is the number of points in the local neighborhood of p_c and μ is the centroid of \mathcal{P}_c . Subsequently, an eigenvalue decomposition of M that results in three orthogonal eigenvectors is performed, which can be used to define an invariant RF. As the SBP descriptor uses a 3D grid to binarize the local neighborhood of p_c , only a pair of orthogonal vectors are needed to define the invariant RF. Thus, the eigenvector corresponding to the smallest eigenvalue is selected. Note that this vector is typically used to approximate the normal to p_c [43]. Due to the second orthogonal vector requirement, we also select the eigenvector with the highest eigenvalue. Their orientation is determined to be coherent with the vectors they represent [6, 50]. Finally, the third vector of the RF can be set to the dot product of the already selected eigenvectors, only to avoid improper rotation transformations (reflections).

Once the two vectors of the RF have been computed, we use their orientations to overlap a 3D grid over \mathcal{P}_c . As shown in Fig. 1, the RF_z vector is always mapped to the upper vector (z -axis) of the 3D grid, and the RF_x to the front one (x -axis). There are two parameters to be set in this grid: the number of bins k , and the bin size l . Based on empirical evidence, we propose the generation of patterns with size $k = 64$, using a grid of $4 \times 4 \times 4$. Regarding the bin size l , as the 3D grid is fitted into the sphere formed by the search radius R (i.e. R corresponds to half the diagonal of a 3D cube of side $l \cdot \sqrt[3]{k}$), l can be calculated as follows:

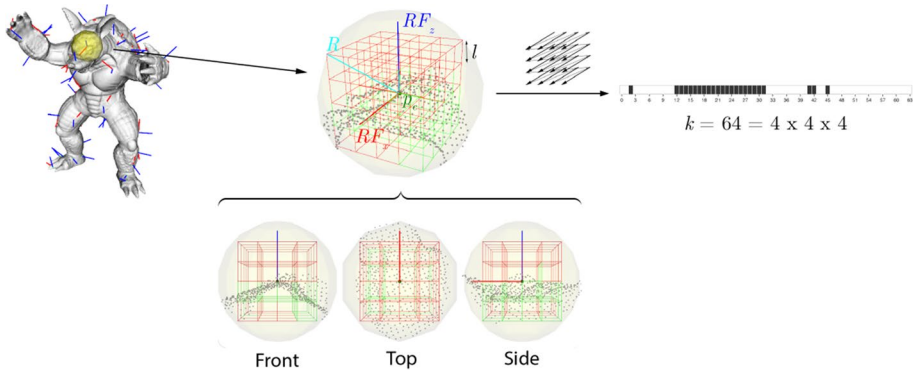


Fig. 1 SBP generation. The 3D grid is overlapped over p_c based on its calculated RF (where the blue vector represents the RF_z and corresponds to the eigenvector with smallest eigenvalue, and the red vector represents RF_x and corresponds to the eigenvector with the highest eigenvalue), and then a pattern is generated based on the appearance of points in the corresponding bin

$$l = \frac{2R}{\sqrt[3]{k} \cdot \sqrt{3}} \tag{4}$$

Finally, each point in \mathcal{P}_c is assigned to the corresponding bin based on its position in the local RF. The binary pattern T that encodes the presence (or absence) of at least one point inside each bin is then created. That is,

$$T = \langle f(b_1), \dots, f(b_K) \rangle, \quad f(x) = \begin{cases} 1 & \text{if } |x| > 0 \\ 0 & \text{if } |x| \leq 0 \end{cases} \tag{5}$$

where T represents the final SBP descriptor, b_i the set of points assigned to the i -th bin, and $f(x)$ is a function that returns a binary value depending on the number of points in the bin. Figure 1 shows an illustration of the process for generating a SBP descriptor.

4 Keypoint detection with SBP

As stated, the LBP proposal includes the definition of uniform patterns [32], which are a subset of all the possible patterns. Specifically, the term “uniform patterns” refers to those patterns that have all their 1s together, in other words, there are no more than two changes between 1s and 0s along the pattern. These uniform patterns are the most frequent and represent basic shapes that can be found in an image.

This approach can be paralleled for point clouds by analyzing the 1s distribution in the SBP patterns. The main idea is to compute the SBP pattern for each point in the input point cloud. Then, based on whether it is a uniform pattern or not, and on its number of 1s, we select it (or not) as keypoint. Later, in Section 5, we will also describe how to optimize the detection process for real-time applications. We can thus distinguish two basic steps to generate a keypoint detector: a) identifying the points corresponding to uniform patterns, and b) selecting the most representative patterns in the image.

4.1 Uniform patterns

In 3D, we determine a uniform pattern by analyzing the disposition of 1s in the 3D grid. Specifically, we consider that a pattern is uniform if all the 1s are in contiguous positions in the 3D grid.

Formally, given the 3D grid disposition of the binary values around an interest point, we consider its pattern as uniform if there is only one connected component of 1s in the pattern. In other words, given the subset \mathcal{B}_1 of bins with value 1, for any two bins $b_i \in \mathcal{B}_1$ and $b_j \in \mathcal{B}_1$, there is a path of 1s between them in the grid (with steps at Manhattan distance of 1).

Based on this criterion, we then assign to each pattern T an index value U_T based on the number of 1s it contains:

$$U_T = \begin{cases} \sum_{i=1}^k f(b_i) & \text{if } T \text{ is uniform} \\ k + 1 & \text{if } T \text{ is not uniform} \end{cases} \quad (6)$$

where k is the size of the pattern (in this case $k = 64$) and $f(x)$ is the function defined in (5) to calculate the binary value of the bins. With this index value, we cluster the patterns according to their number of 1s, and independently of their distribution in the final pattern.

4.2 Selection criteria

After assigning a U_T to each interest point in the image, we proceed to select the most representative subset of points among those corresponding to uniform patterns. First, we generate a histogram of the index assigned to the patterns and we then select the final keypoints using one of the following criteria:

- Frequency (F_n): The points with the n less frequent U_T , with $n \in [1, k]$.
- Minimum U_T (m_n): The minimum index value of uniform pattern, that is, we select all points with pattern T if $U_T \geq n$ with $n \in [1, k]$.
- Number of pattern indices (N_n): The $n/2$ with the lowest index value and the $n/2$ with the highest index value, that is, we select all points with pattern T if $U_T \leq n/2$ or $U_T \geq k - n/2$ with $n \in [1, k]$.
- Number of points (M_m): We select the points with the least frequent U_T while there are fewer than m points selected, with $m > 0$.

In Fig. 2 we show an example of keypoints detected using different selection criteria.

5 Experimental results

In this section, we use the evaluation benchmark proposed in [45] to assess the performance of our proposals. This benchmark contains five different datasets to compare against state-of-the-art techniques for two main applications: keypoint detection and local descriptor generation. Each dataset contains mesh files for a set of models $\mathcal{M} = \{\mathbf{M}_1, \dots, \mathbf{M}_M\}$ and scenes $\mathcal{S} = \{\mathbf{S}_1, \dots, \mathbf{S}_S\}$, where each scene is composed of a subset of the models. Additionally, each dataset includes the ground-truth

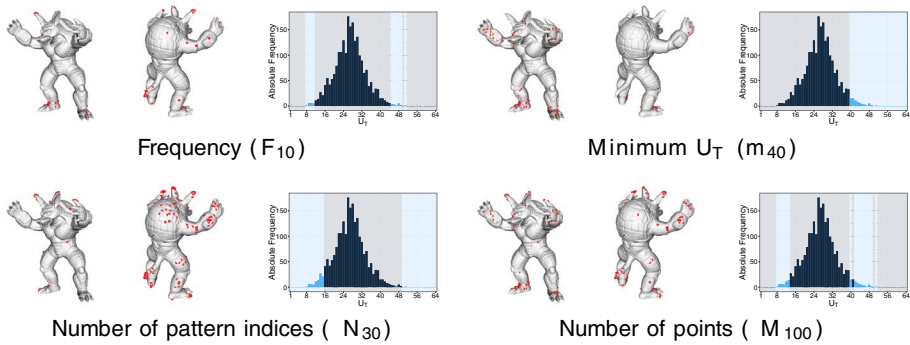


Fig. 2 Example of keypoints based on selection criteria. The light color in the histogram represents the selected U_T

transformations (\mathbf{T}_{ms}) to overlap each model \mathbf{M}_m with its instance in the scene \mathbf{S}_s . Specifically, the datasets are:

- *Retrieval*: a synthetic dataset containing 3D models of 6 objects and 3D models of 18 scenes at different noise levels.
- *Random Views*: a synthetic dataset containing 3D models of 6 objects and 2.5D views of 36 scenes at different noise levels.
- *Laser Scanner* [26]: a dataset captured with a Minolta Vivid 910 scanner that contains 3D models of 5 objects and 2.5D views of 50 scenes.
- *Space Time*: this contains 2.5D views of 6 objects and 2.5D views of 12 scenes.
- *Kinect*: a dataset captured with a Microsoft Kinect device containing 2.5D views of 6 objects and 2.5D views of 12 scenes.

In Fig. 3, we show a sample model and three scenes for each dataset.

Finally, it should be noted that both our proposal and the methods selected for comparison are based exclusively on the local information around the interest point. In 3D, the local neighborhood includes the points in a sphere around the keypoint with radius R_{mesh} resolution (mr), where the mesh resolution is defined as the mean length of the edges in the input object mesh. In the following, we will refer to the value of R as scale (it represents a scale of the mesh resolution).

For simplicity, in this section, we will only discuss the results obtained with the *Random Views* dataset, as it presents a typical scenario in object recognition and manipulation, that is, the identification of the partial view of an object using its 3D model. However, in Appendix 1, we include the results with the other datasets.

The source code for these experiments was implemented using the Point Cloud Library (PCL [44])¹.

¹ Source code available at: <http://simdresearch.com/supplements/sbp/>



Fig. 3 Sample of one model and three scenes for each dataset. From top to bottom: *Retrieval*, *Random Views*, *Laser Scanner*, *Space Time*, and *Kinect*. The first two datasets show the same scene at different noise levels; the other three datasets show different scenes

5.1 Keypoint detector evaluation

To evaluate the performance of a keypoint detector, it is necessary to measure its repeatability [46], that is, the same keypoints are selected in different instances of the same object. Specifically, we use the repeatability measures proposed in [45], which are defined

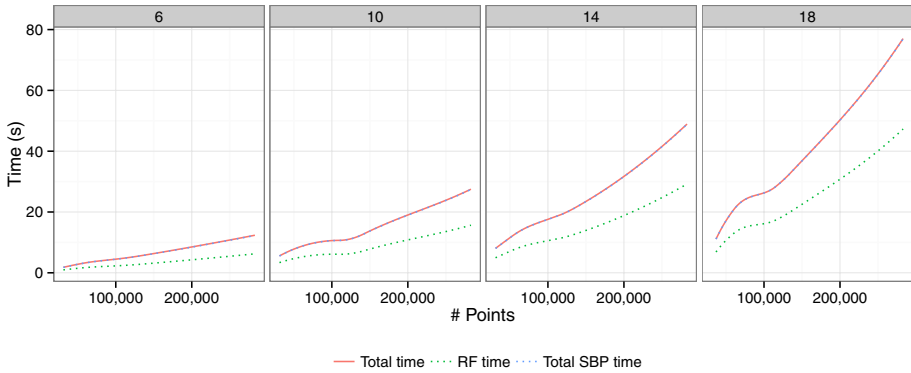


Fig. 4 Time per point cloud size in SBP keypoint calculation

as follows: given a keypoint k_m^i extracted from the model M_m , it is said to be repeatable if, after its transformation based on the ground-truth information T_{ms} , the Euclidean distance to its nearest neighbor k_s^j in the set of keypoints extracted from the scene S_s is less than a threshold $\epsilon = 2$ mr.

Thus, given the set RK_{ms} of repeatable keypoints for the model M_m in the scene S_s , the absolute repeatability is defined as

$$r_{abs} = |RK_{ms}| \tag{7}$$

and the relative repeatability as

$$r_{rel} = \frac{|RK_{ms}|}{|VK_{ms}|} \tag{8}$$

where $|VK_{ms}|$ is the subset of keypoints from the model M_m that are visible in the scene S_s . A keypoint is considered to be visible if, after applying its ground-truth transformation to the model, there is a point in the scene in a small local neighborhood (2 mr) around the transformed object keypoint.

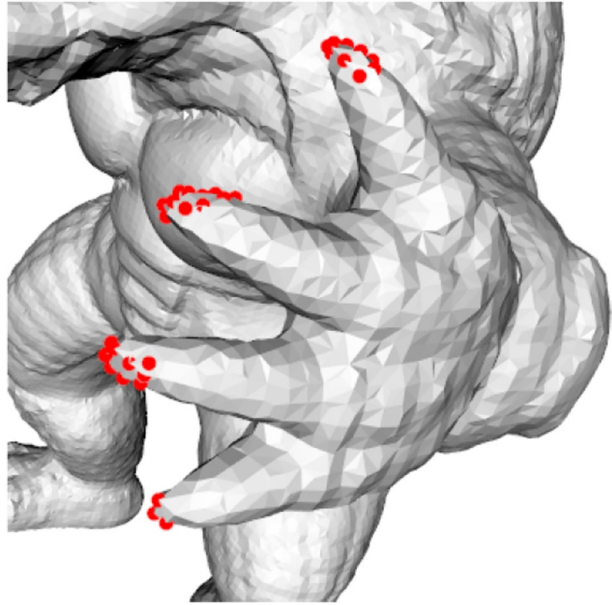
In Section 4.2, we proposed different methods to select the final keypoints based on the distribution of uniform patterns in the images. First, therefore, we will evaluate the performance of the different options proposed to determine the best configuration. We will then compare that configuration against state-of-the-art detectors.

5.1.1 Optimization for real-time applications

The keypoint detector presented thus far is intended to identify relevant basic shapes given an input image. However, it has two major drawbacks that need to be addressed:

1. The detection of uniform patterns entails the computation of the local Reference Frame for each point. This stage is the most computationally expensive step in the SBP generation. While this is an assumable cost for generating rotation invariant descriptors, it is not efficient enough for real-time applications. In Fig. 4, we can see the time employed to select keypoints, calculating the SBP descriptor for all the points in the input point

Fig. 5 Example of concentration of selected points with the same U_T



- cloud. As expected, these times are excessively high for real-time applications and most of the time is spent in calculating the RF and transforming the points to determine their position in the pattern grid.
- Given a point p_c , the SBP descriptor uses a 3D grid to encode the presence of points in its local neighborhood. This grid has a bin size of l , which is determined by the radius R used to select the points that belong to that local neighborhood. This means that points with a distance of less than l will probably be described with the same SBP pattern and the same U_T and, consequently, all these points will be selected simultaneously, as the deciding factor to detect a keypoint is the number of 1s in its SBP representation (see Fig. 5).

We can address these issues by removing the calculation of the local Reference Frame. If we skip this step, we should still be able to identify basic shapes, as they would have similar but rotated patterns. We now directly discretize the whole input image by overlapping a 3D grid of bin size l and computing the presence of points in each bin. Then, we analyze the distribution of uniform patterns over this 3D grid of the input point cloud, and select the interest U_T based on our selection criteria. Finally, to avoid the concentration of points belonging to the same U_T , we select the closest point to the grid center of the pattern as the keypoint, as it would be the most feasible point to present the selected U_T .

We need to validate whether this change would affect the repeatability of the SBP keypoint detector. To this end, we evaluate the relative repeatability of both approaches for the SBP keypoint detector with the same configuration. That is, given a keypoint k_s^i extracted from the scene S_s , using the SBP keypoint detector with RF calculation, and the closest keypoint k_s^{ij} extracted from the same scene S_s , using the same SBP keypoint detection method but without calculating the RF, we will consider the keypoint k_s^i repeatable if $|k_s^i - k_s^{ij}| < 2 \text{ mr}$.

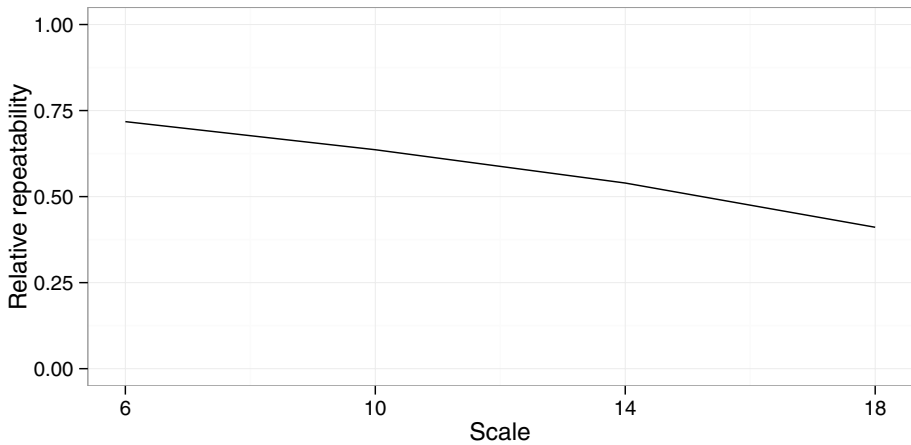


Fig. 6 Repeatability of SBP keypoint detector without calculating the local RF

In Fig. 6, we can observe the relative repeatability of the keypoint detector when optimized for real-time applications. This figure showcases the high repeatability of keypoints when we remove the RF calculation at lower scales. At higher scales, the repeatability diminishes due to the separation that we intentionally include in the SBP keypoint detection to avoid concentration of points with the same U_T (see Fig. 5). In Section 5.1.3, we will demonstrate that these results are still competitive against state-of-the-art detectors.

5.1.2 Keypoint selection criteria

Ideally, a keypoint detector should generate a small (but considerable) number of points that can always be identified in different instances of an object. That is, a good keypoint detector should present a high relative repeatability without selecting an excessive number of points. Thus, to compare the different configurations of the SBP keypoint detector, we define the repeatability score r_{score} as a weighted harmonic mean of the relative repeatability r_{rel} and the absolute repeatability r_{abs} :

$$r_{score} = \frac{4}{\frac{3}{-n(r_{abs})+1} + \frac{1}{n(r_{rel})}} \quad (9)$$

where $n(r)$ is the normalized value of the repeatability measure r . This normalization is performed due to the different ranges of both r_{abs} and r_{rel} .

In Fig. 7, we can observe the normalized r_{abs} and r_{rel} for all the possible configurations in each dataset, and the best configuration based on its r_{score} . As expected, the selected configurations are those that maximize r_{rel} and minimize the r_{abs} .

In Table 1, we show the best configuration for each dataset based on its r_{score} . From these results, we can conclude that the best configuration is N_{30} , as it obtains the best score with two of the datasets, *Random Views* and *Laser Scanner*. Additionally, the best configuration for the Space Time dataset is N_{32} , which is remarkably similar to the previous one, reinforcing the selection of the overall best configuration (N_{30}) for the following experiments.

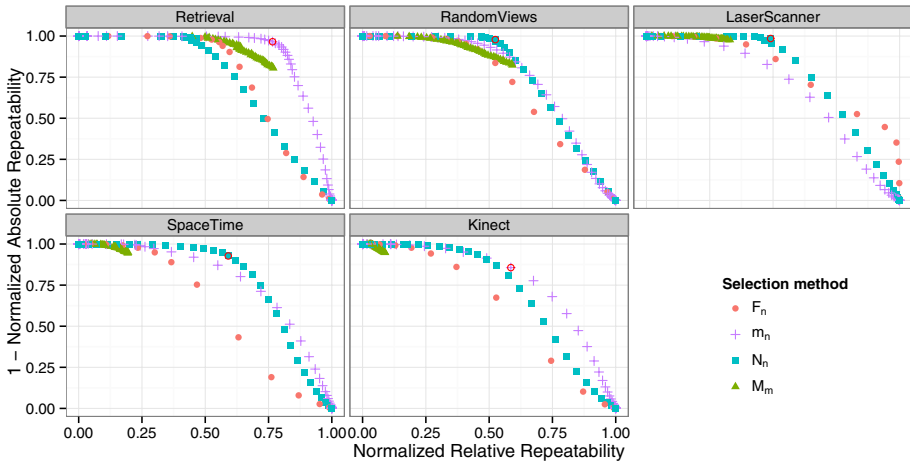


Fig. 7 Normalized absolute and relative repeatabilities by dataset and selection method. The highlighted point represents the selected configuration based on its score (r_{score})

Table 1 Best selection method and its score by dataset

Dataset	Best configuration	r_{score}
<i>Retrieval</i>	m_{38}	0.906317
<i>Random Views</i>	N_{30}	0.804230
<i>Laser Scanner</i>	N_{30}	0.786823
<i>Space Time</i>	N_{32}	0.811766
<i>Kinect</i>	m_{26}	0.767785

5.1.3 Comparison with the state of the art

The best configuration of the SBP keypoint detector (N_{30}) was compared against well-known techniques implemented in the PCL library. Specifically, we selected ISS and Harris 3D. Other techniques were discarded because they require color/grayscale information (SIFT 3D), or because they work with specific input images that are unavailable in the selected datasets (for example, the NARF keypoint detector uses range images as input).

The relative and absolute repeatabilities using the *Random Views* dataset are shown in Fig. 8. Here, we can observe that the SBP keypoint detector is better than the state-of-the-art methods ISS and Harris 3D, as it offers a slightly better r_{rel} without selecting too many points. These results are consistently independent of the noise level and scale. Then, in Fig. 9 we show the computation time per input point for different scales and noise levels. It is worth noting that the the SBP keypoint detector offers a uniform detection time which is always lower than for the real-value detectors.

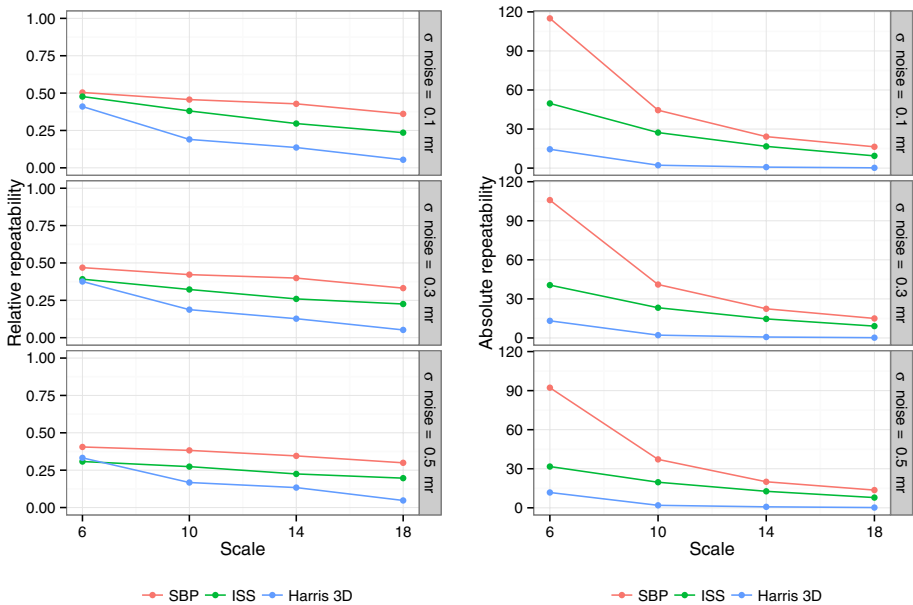


Fig. 8 Absolute and relative repeatability on the *Random Views* dataset, for different scales and noise levels

5.2 Descriptor evaluation

Descriptor evaluation is usually performed based on the 1 – *precision vs recall* curve of the matches for each pair model-scene [27]. For each keypoint k_m^i from the model M_m and each keypoint k_s^j from the scene S_s , we extract their corresponding descriptors, d_m^i and d_s^j , respectively. We subsequently perform an approximated nearest neighbor search [30, 31] between the set of descriptors from the scene and the set of descriptors from the model. For each keypoint k_s^j in the scene, we will consider the ratio between the descriptor distance to its closest keypoint k_m^i in the model ($\|d_s^j - d_m^i\|$) and its second closest keypoint $k_m^{i'}$ ($\|d_s^j - d_m^{i'}\|$), to determine the matches between both images. k_s^j and k_m^i will be considered a match, if this ratio is less than a threshold τ .

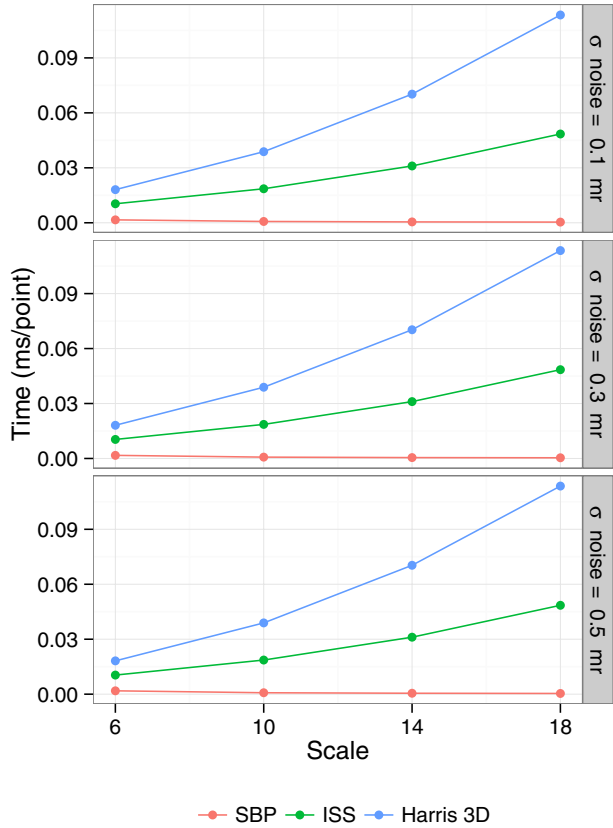
$$\frac{\|d_s^j - d_m^i\|}{\|d_s^j - d_m^{i'}\|} < \tau \tag{10}$$

Finally, and similarly to the keypoint evaluation, a match will be considered as correct if, after its transformation based on the ground-truth information T_{ms} , the Euclidean distance between the model keypoint k_m^i and the scene keypoint k_s^j is less than a threshold $\epsilon = 2$ mr.

In this matching scenario, the *precision* is the fraction of matches that are correct, while the *recall* is the fraction of correct matches retrieved.

$$precision = \frac{\# \text{ correct matches}}{\# \text{ matches}} \tag{11}$$

Fig. 9 Time per point in the input cloud on the *Random Views* dataset, for different scales and noise levels



$$recall = \frac{\# \text{ correct matches}}{\# \text{ correspondences}} \tag{12}$$

We will perform this evaluation against the local descriptors SHOT, FPFH, and Spin Images (SI) that are implemented in the PCL library. Other local descriptors have been discarded for different reasons, for example, the use of color information (CSHOT [51]), specific types of images (NARF uses range images), or because there are newer but similar versions of these base descriptors (PFH [41]). Finally, in order to avoid any bias caused by the keypoint detection method, we randomly select 1000 feature points from each scene, and then calculate the corresponding points in the models based on the ground truth transformations.

Additionally, we will also perform this evaluation against DIP [34] descriptors, which are an example of local descriptors generated using modern Deep Learning techniques. Unlike handcrafted features, these Deep Learning based methods require a high number of images to train. So we cannot rely exclusively on the datasets selected for this study if we intend to train and evaluate Deep Learning methods effectively. The DIP descriptors used in our experiments have been pretrained using the 3DMatch dataset [57].

In Fig. 10, we show the precision-recall curve with the *Random Views* dataset at different noise levels and scales. The best results are obtained with DIP, which is expected because it is the only evaluated technique based on Deep Learning. Among the handcrafted

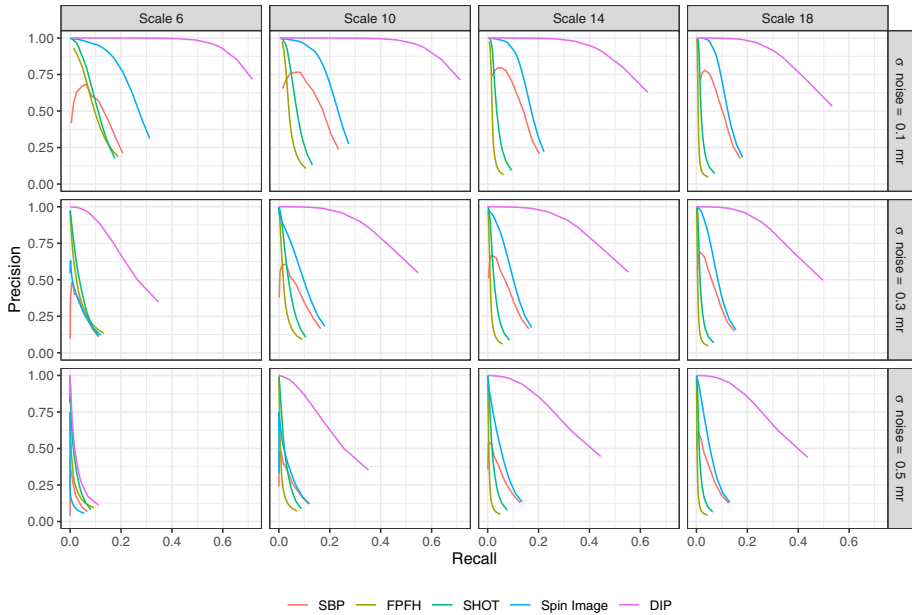


Fig. 10 Matching precision-recall on the *Random Views* dataset, for different scales and noise levels

features, the best results are obtained with SI. However, SBP outperforms FPFH and SHOT, which usually have a low recall. From these results, we can conclude that SBP shows competitive results against handcrafted descriptors, but Deep Learning solutions are preferred when sufficient computational resources are available.

Then, in Fig. 11 we show the computation time per input point for different scales and noise levels. Again, it is worth noting the stable generation and matching computational time of the SBP descriptor, which clearly improves real-value descriptors. We have not included DIP descriptors in this time comparison because the resources needed to generate those descriptors in a reasonable time are much higher than the other methods. So, even if they have a remarkable performance in terms of precision and recall, these descriptors could only be used in real-time applications in platforms with, at least, a dedicated GPU and a high-end CPU that is able to calculate the local reference frame needed for each descriptor as fast as possible in order to avoid bottlenecks.

Finally, another requirement to consider is the memory footprint of the different descriptors. Table 2 contains the length and size in bytes of the different descriptors. The SBP descriptor is 16 times smaller than **DIP**, the smallest of the real-valued descriptors considered.

6 Conclusions

In this paper, we have presented a binary descriptor and keypoint detector for point clouds, and more specifically, for 3D vision applications that require an efficient and low-cost processing. Our proposals can be easily integrated in any robot vision application (see Fig. 12).

Fig. 11 Time per correspondence on the *Random Views* dataset, for different scales and noise levels

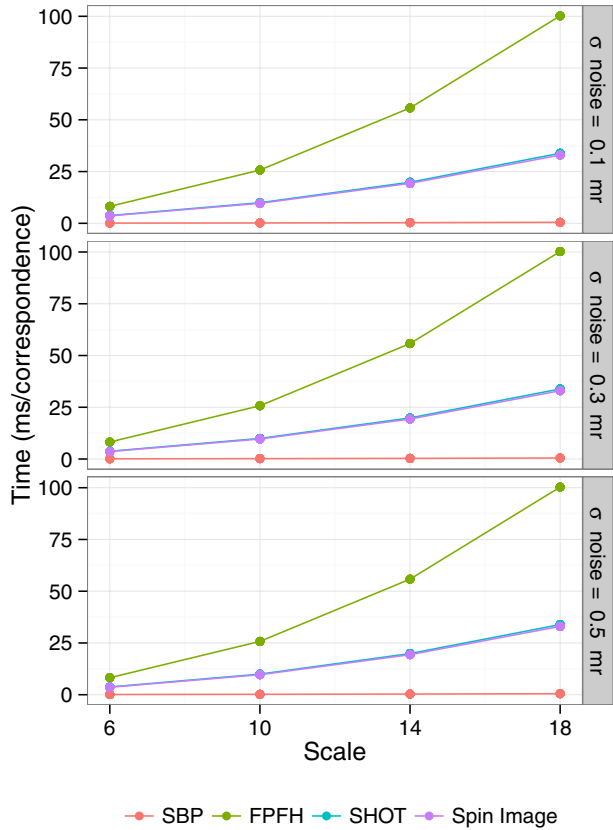


Table 2 Descriptor size

Descriptor	Type	Length	Size (Bytes)
SBP	Binary	64	8*
DIP	Real	32	128
FPFH	Real	33	132
SHOT	Real	352	1408
SI	Real	153	612

The size in bytes is calculated considering that a Boolean value is stored in 1 bit and a real value is stored in 4 bytes

* Depending on the implementation, a Boolean might be stored in 1 byte; and, in that case, the SBP descriptor would use 64 bytes

To assess the performance of our proposals, we selected a well-known evaluation benchmark that includes different datasets, and we compared our methods against the state-of-the-art techniques implemented in the PCL library. The results show that both the SBP descriptor and the SBP keypoint detector perform similarly in terms of precision-recall and repeatability, respectively, to these well-known techniques. However, the methods here proposed are significantly better in computation time and memory footprint.

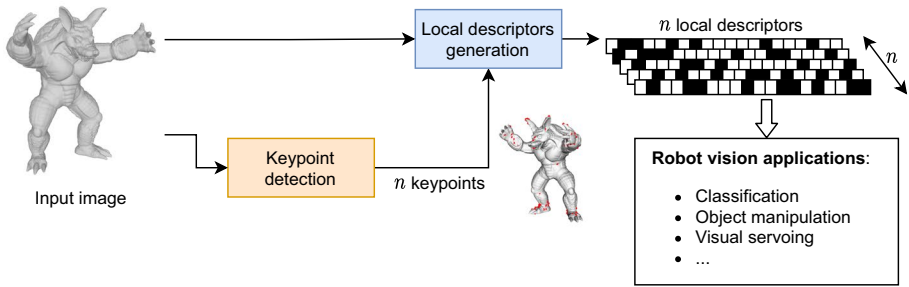


Fig. 12 Flowchart that shows how to use the techniques described in this paper

The proven efficiency of the SBP methods makes them the ideal choice for real-time applications and systems with low computational requirements. For example, robot vision tasks that must be run in a robot with a single processor might greatly benefit from these efficient approaches based on binary values. However, if these computational aspects are of no concern, and based on the results obtained, modern approaches based on Deep Learning should also be considered as alternative solutions.

Additionally, the work here presented could be extended in the future to use color information, for example, by encoding and binarizing the mean color of the points in one bin of the 3D grid. This type of extension should provide better descriptiveness to improve the matching capabilities of the SBP descriptor.

Appendix A: Additional experiments

This Appendix includes the results from the experiments depicted on Section 5 with the datasets: *Retrieval*, *Laser Scanner*, *Space Time*, and *Kinect*.

A.1 Keypoint repeatability

The absolute and relative repeatabilities at different scales and noise levels are shown in Fig. 13 for the *Retrieval* dataset. In this particular instance, the repeatability of the SBP detector is lower than the state-of-the-art proposals. This behavior is probably caused by the simplification that removes the calculation of an invariant RF, which seems to be essential for this specific dataset. However, with the other datasets (see Fig. 14), the behavior of the detectors is similar to that obtained with the *Random Views* dataset; SBP has a similar relative repeatability to ISS and Harris 3D without the need to select a high number of keypoints.

Figure 15 shows the mean time to detect keypoints per point in the input cloud. In this case, the results are similar to those obtained with the *Random Views* dataset. In view of these results, we can state that SBP outperforms the state-of-the-art detectors.

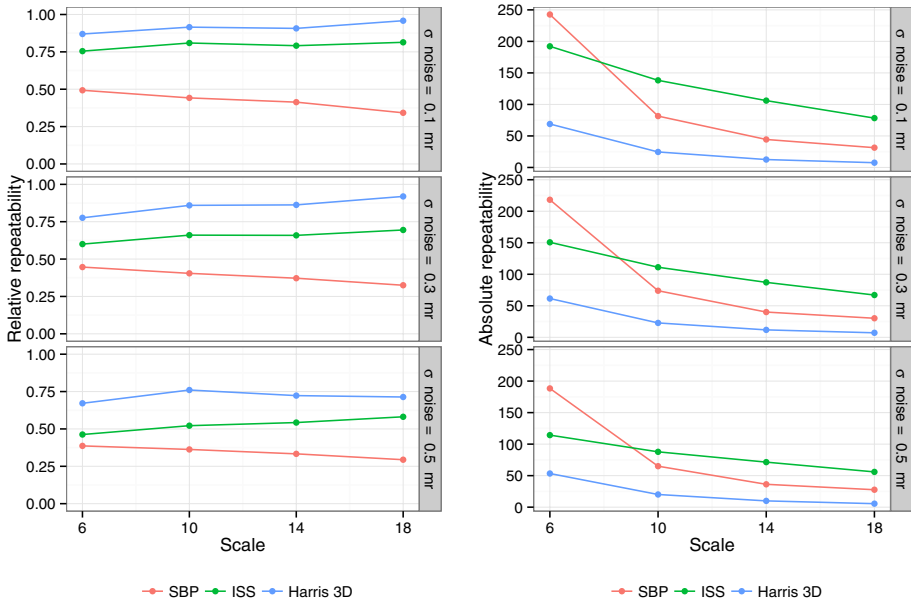


Fig. 13 Absolute and relative repeatabilities on the *Retrieval* dataset, for different scales and noise levels

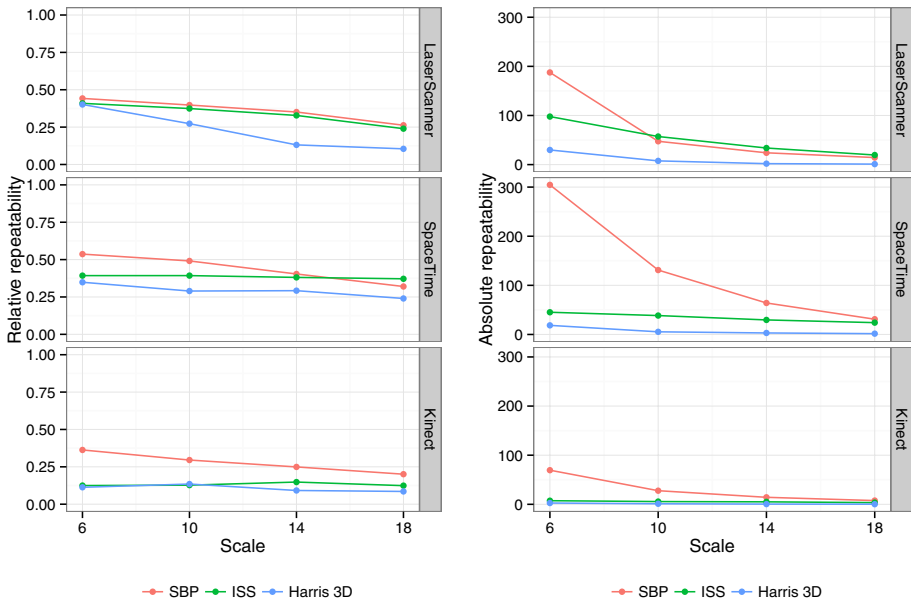


Fig. 14 Absolute and relative repeatabilities on the *Laser Scanner*, *Space Time*, and *Kinect* datasets, for different scales

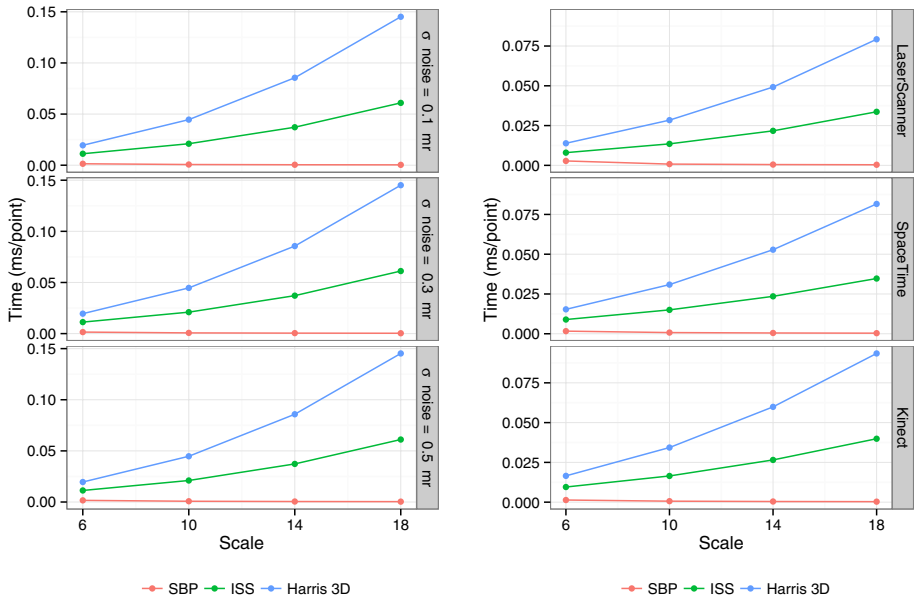


Fig. 15 Time per point in the input cloud on the *Retrieval* dataset (left) and the *Laser Scanner*, *Space Time*, and *Kinect* datasets (right), for different scales

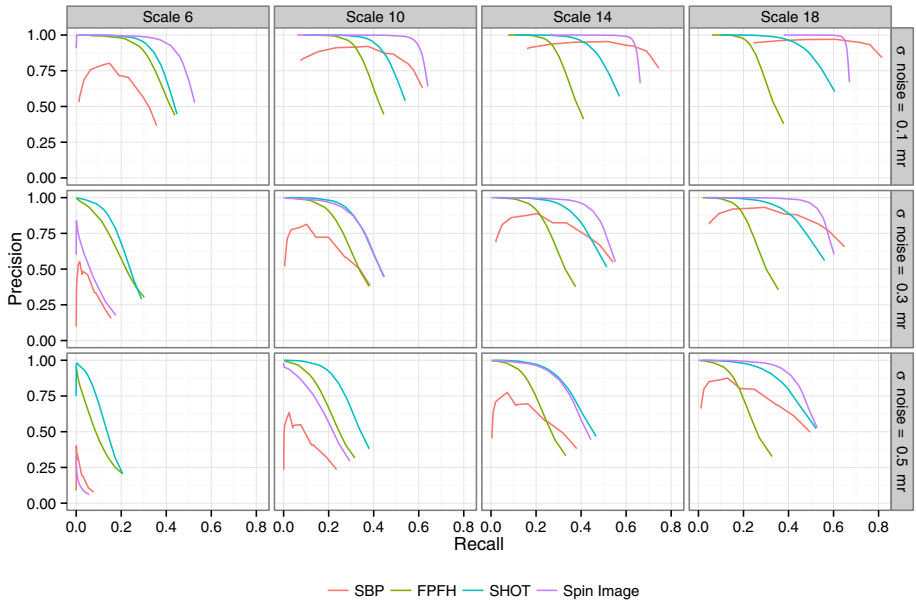


Fig. 16 Matching precision-recall on the *Retrieval* dataset, for different scales and noise levels

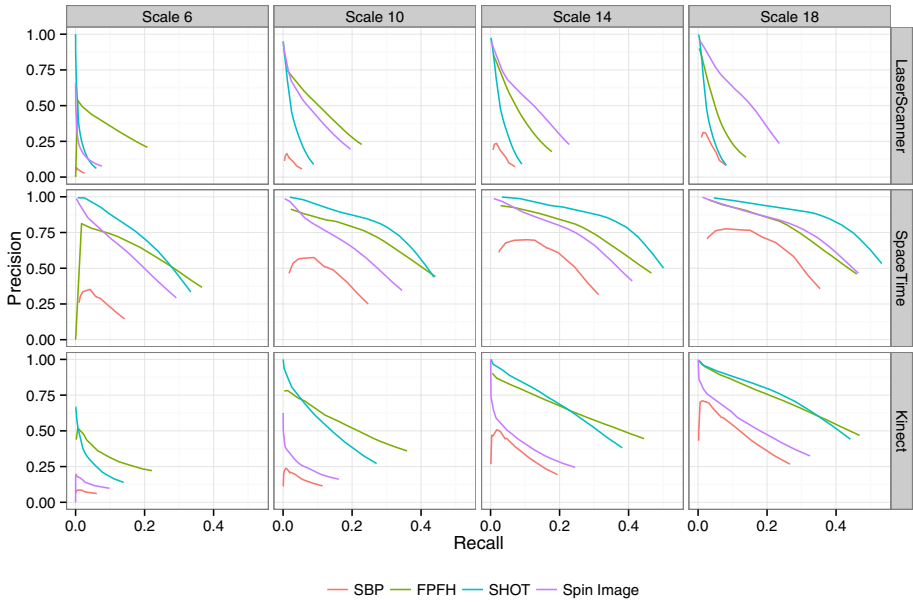


Fig. 17 Matching precision-recall on the *Laser Scanner*, *Space Time*, and *Kinect* datasets, for different scales

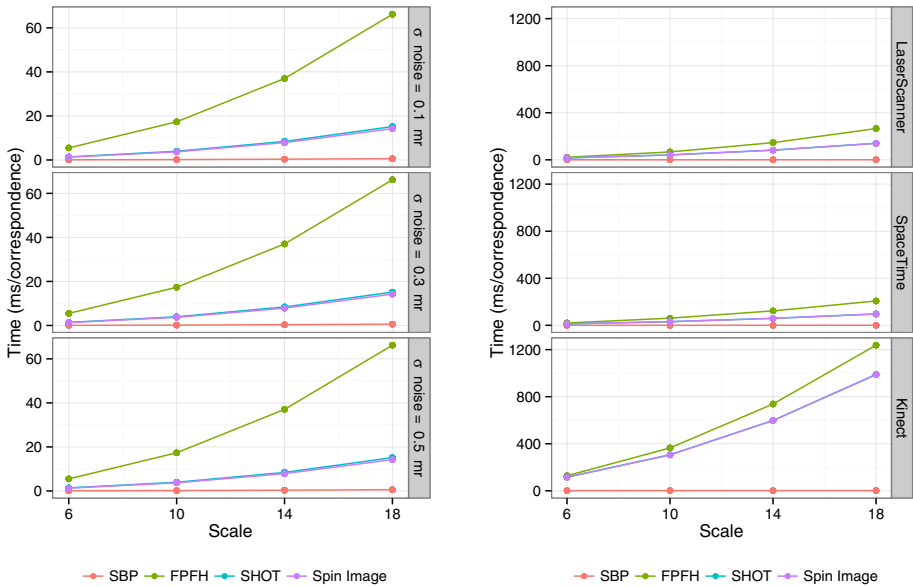


Fig. 18 Time per correspondence with the *Retrieval* dataset (left) and the *Laser Scanner*, *Space Time*, and *Kinect* datasets (right), for different scales

A.2 Descriptor evaluation

The precision and recall at different noise levels are shown in Fig. 16 for the *Retrieval* dataset. For lower scales, state-of-the-art descriptors outperform SBP. This is probably related to the grid representation in the SBP descriptor, where lower scales result in bins too small to actually encode the shape around the interest point. However, at high scales, SBP improves considerably and even obtains better results than the real-value descriptors. When comparing with the other datasets (see Fig. 17), SBP has a slightly worse behavior than state-of-the-art descriptors, but competitive enough depending on the application due to its fast computation and small memory footprint.

Figure 18 shows the mean time to generate the descriptors and perform the matching per correspondence. In this case, the results are similar to those obtained with the *Random Views* dataset. SBP outperforms the state-of-the-art descriptors.

Acknowledgements This work has been funded by the Government of Castilla-La Mancha and "ERDF A way of making Europe" through the project SBPLY/17/180501/000493. It is also part of the project PID2019--106758GB--C33 funded by MCIN/AEI/ 10.13039/501100011033.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Ahonen T, Hadid A, Pietikainen M (2006) Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28(12):2037–2041. <https://doi.org/10.1109/TPAMI.2006.244>
2. Alahi A, Ortiz R, Vanderghenst P (2012) FREAK: fast retina keypoint. In: *IEEE conference on computer vision and pattern recognition (CVPR)*, pp 510–517. <https://doi.org/10.1109/CVPR.2012.6247715>
3. Alexandre LA (2012) 3D descriptors for object and category recognition: a comparative evaluation. In: *Workshop on color-depth camera fusion in robotics at the IEEE/RSJ international conference on intelligent robots and systems (IROS workshops)*
4. Bay H, Tuytelaars T, Van Gool L (2006) SURF: speeded up robust features. In: *Computer vision – ECCV 2006, Lecture notes in computer science*, vol 3951. Springer, Berlin, pp 404–417. https://doi.org/10.1007/11744023_32
5. Bo L, Ren X, Fox D (2011) Depth kernel descriptors for object recognition. In: *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp 821–826. <https://doi.org/10.1109/IROS.2011.6095119>
6. Bro R, Acar E, Kolda TG (2008) Resolving the sign ambiguity in the singular value decomposition. *Journal of Chemometrics* 22(2):135–140. <https://doi.org/10.1002/cem.1122>
7. Calonder M, Lepetit V, Strecha C, Fua P (2010) BRIEF: binary robust independent elementary features. In: *Computer vision – ECCV 2010, Lecture notes in computer science*, vol 6314. Springer, Heidelberg, pp 778–792. https://doi.org/10.1007/978-3-642-15561-1_56

8. Canclini A, Cesana M, Redondi A, Tagliasacchi M, Ascenso J, Cilla R (2013) Evaluation of low-complexity visual feature detectors and descriptors. In: 18th international conference on digital signal processing (DSP), pp 1–7. <https://doi.org/10.1109/ICDSP.2013.6622757>
9. Castellani U, Cristani M, Fantoni S, Murino V (2008) Sparse points matching by combining 3d mesh saliency with statistical descriptors. *Computer Graphics Forum* 27(2):643–652. <https://doi.org/10.1111/j.1467-8659.2008.01162.x>
10. Chen H, Bhanu B (2007) 3D free-form object recognition in range images using local surface patches. *Pattern Recognition Letters* 28(10):1252–1262. <https://doi.org/10.1016/j.patrec.2007.02.009>
11. Deng H, Birdal T, Ilic S (2018) Ppfnet: Global context aware local features for robust 3d point matching. In: IEEE conference on computer vision and pattern recognition (CVPR)
12. Fehr J (2007) Rotational invariant uniform local binary patterns for full 3d volume texture analysis. In: Finnish signal processing symposium (FINSIG)
13. Fehr J, Burkhardt H (2008) 3D rotation invariant local binary patterns. In: 19th international conference on pattern recognition (ICPR), pp 1–4. <https://doi.org/10.1109/ICPR.2008.4761098>
14. Guo Y, Wang H, Hu Q, Liu H, Liu L, Bennamoun M (2020) Deep learning for 3d point clouds: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* :1–1. <https://doi.org/10.1109/TPAMI.2020.3005434>
15. Guo Z, Zhang D, Zhang D (2010) A Completed Modeling of Local Binary Pattern Operator for Texture Classification. *IEEE Transactions on Image Processing* 19(6):1657–1663. <https://doi.org/10.1109/TIP.2010.2044957>
16. Huang X, Li S, Wang Y (2004) Shape localization based on statistical method using extended local binary pattern. In: IEEE first symposium on multi-agent security and survivability, pp 184–187. <https://doi.org/10.1109/ICIG.2004.127>
17. Johnson AE, Hebert M (1999) Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21(5):433–449. <https://doi.org/10.1109/34.765655>
18. Koenderink JJ, van Doorn AJ (1992) Surface shape and curvature scales. *Image and Vision Computing* 10(8):557–564. [https://doi.org/10.1016/0262-8856\(92\)90076-F](https://doi.org/10.1016/0262-8856(92)90076-F)
19. Krainin M, Curless B, Fox D (2011) Autonomous generation of complete 3D object models using next best view manipulation planning. In: IEEE international conference on robotics and automation (ICRA), pp 5031–5037. <https://doi.org/10.1109/ICRA.2011.5980429>
20. Krizhevsky A, Sutskever I, Hinton GE (2017) Imagenet classification with deep convolutional neural networks. *Commun ACM* 60(6):84–90. <https://doi.org/10.1145/3065386>
21. Leutenegger S, Chli M, Siegwart R (2011) BRISK: binary robust invariant scalable keypoints. In: IEEE International conference on computer vision (ICCV), pp 2548–2555. <https://doi.org/10.1109/ICCV.2011.6126542>
22. Lowe DG (2004) Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* 60(2):91–110. <https://doi.org/10.1023/B:VISI.0000029664.99615.94>
23. Mair E, Hager G, Burschka D, Suppa M, Hirzinger G (2010) Adaptive and generic corner detection based on the accelerated segment test. In: Computer vision – ECCV 2010, Lecture notes in computer science, vol 6312. Springer, Berlin, pp 183–196. https://doi.org/10.1007/978-3-642-15552-9_14
24. Martínez-Carranza J, Mayol-Cuevas W (2013) Real-time continuous 6d relocalisation for depth cameras. In: Workshop on multi-view geometry in robotics (MVGRO)
25. Martínez-Gómez J, Fernández-Caballero A, García-Varea I, Rodríguez L, Romero-González C (2014) A taxonomy of vision systems for ground mobile robots. *International Journal of Advanced Robotic Systems* 11
26. Mian A, Bennamoun M, Owens R (2010) On the Repeatability and Quality of Keypoints for Local Feature-based 3D Object Retrieval from Cluttered Scenes. *International Journal of Computer Vision* 89(2–3):348–361. <https://doi.org/10.1007/s11263-009-0296-z>
27. Mikolajczyk K, Schmid C (2005) A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27(10):1615–1630. <https://doi.org/10.1109/TPAMI.2005.188>
28. Miksik O, Mikolajczyk K (2012) Evaluation of local detectors and descriptors for fast feature matching. In: 21st international conference on pattern recognition (ICPR), pp 2681–2684
29. Morgado P, Silveira M, Marques J (2013) Extending local binary patterns to 3D for the diagnosis of Alzheimer’s disease. In: IEEE 10th international symposium on biomedical imaging (ISBI 2013), pp 117–120. <https://doi.org/10.1109/ISBI.2013.6556426>
30. Muja M, Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. In: International conference on computer vision theory and application (VISSAPP), pp 331–340

31. Muja M, Lowe DG (2012) Fast matching of binary features. In: Ninth conference on computer and robot vision (CRV). pp 404–410. <https://doi.org/10.1109/CRV.2012.60>
32. Ojala T, Pietikainen M, Maenpaa T (2002) Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24(7):971–987. <https://doi.org/10.1109/TPAMI.2002.1017623>
33. Paulhac L, Makris P, Ramel JY (2008) Comparison between 2D and 3D local binary pattern methods for characterisation of three-dimensional textures. In: Image analysis and recognition, Lecture notes in computer science, vol 5112. Springer, Berlin, pp 670–679. https://doi.org/10.1007/978-3-540-69812-8_66
34. Poiesi F, Boscaini D (2021) Distinctive 3d local deep descriptors. In: 2020 25th International conference on pattern recognition (ICPR). pp 5720–5727. <https://doi.org/10.1109/ICPR48806.2021.9411978>
35. Prakhya SM, Liu B, Lin W, Jakhetiya V, Guntuku SC (2017) B-shot: a binary 3d feature descriptor for fast keypoint matching on 3d point clouds. *Autonomous Robots* 41(7):1501–1520. <https://doi.org/10.1007/s10514-016-9612-y>
36. Qi CR, Su H, Mo K, Guibas LJ (2017) Pointnet: Deep learning on point sets for 3d classification and segmentation. In: IEEE conference on computer vision and pattern recognition (CVPR)
37. Romero-González C, Martínez-Gómez J, García-Varea I, Rodríguez-Ruiz L (2016) Binary patterns for shape description in RGB-D object registration. In: IEEE winter conference on applications of computer vision (WACV). pp 1–9. <https://doi.org/10.1109/WACV.2016.7477727>
38. Romero-González C, Martínez-Gómez J, García-Varea I, Rodríguez-Ruiz L (2016) Keypoint detection in RGB-D images using binary patterns. In: Robot 2015: second Iberian robotics conference: advances in robotics, vol 2. Springer International Publishing, pp 685–694. https://doi.org/10.1007/978-3-319-27149-1_53
39. Rosten E, Porter R, Drummond T (2010) Faster and Better: A Machine Learning Approach to Corner Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32(1):105–119. <https://doi.org/10.1109/TPAMI.2008.275>
40. Rublee E, Rabaud V, Konolige K, Bradski G (2011) ORB: An efficient alternative to SIFT or SURF. In: IEEE international conference on computer vision (ICCV). pp 2564–2571. <https://doi.org/10.1109/ICCV.2011.6126544>
41. Rusu R, Marton Z, Blodow N, Beetz M (2008) Learning Informative Point Classes for the Acquisition of Object Model Maps. In: 10th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp 643–650. <https://doi.org/10.1109/ICARCV.2008.4795593>
42. Rusu R, Blodow N, Beetz M (2009) Fast point feature histograms (fpfh) for 3d registration. In: IEEE international conference on robotics and automation (ICRA). pp 3212–3217. <https://doi.org/10.1109/ROBOT.2009.5152473>
43. Rusu RB (2009) Semantic 3D object maps for everyday manipulation in human living environments. PhD thesis, Computer Science department, Technische Universitaet Muenchen, Germany
44. Rusu RB, Cousins S (2011) 3D is here: point cloud library (PCL). In: IEEE international conference on robotics and automation (ICRA), Shanghai, China
45. Salti S, Tombari F, Di Stefano L (2011) A performance evaluation of 3D keypoint detectors. In: 2011 international conference on 3D imaging, modeling, processing, visualization and transmission (3DIMPVT). pp 236–243. <https://doi.org/10.1109/3DIMPVT.2011.37>
46. Schmid C, Mohr R, Bauckhage C (2000) Evaluation of interest point detectors. *International Journal of Computer Vision* 37(2):151–172. <https://doi.org/10.1023/A:1008199403446>
47. Somanath G, Kambhamettu C (2011) Abstraction and generalization of 3D structure for recognition in large intra-class variation. In: Computer vision – ACCV 2010, Lecture notes in computer science, vol 6494. Springer, Berlin, pp 483–496. https://doi.org/10.1007/978-3-642-19318-7_38
48. Steder B, Rusu R, Konolige K, Burgard W (2011) Point feature extraction on 3D range scans taking into account object boundaries. In: IEEE international conference on robotics and automation (ICRA), pp 2601–2608. <https://doi.org/10.1109/ICRA.2011.5980187>
49. Stückler J, Steffens R, Holz D, Behnke S (2013) Efficient 3D object perception and grasp planning for mobile manipulation in domestic environments. *Robotics and Autonomous Systems* 61(10):1106–1115. <https://doi.org/10.1016/j.robot.2012.08.003>
50. Tombari F, Salti S, Di Stefano L (2010) Unique signatures of histograms for local surface description. In: Computer vision – ECCV 2010, Lecture notes in computer science, vol 6313. Springer, Berlin, pp 356–369. https://doi.org/10.1007/978-3-642-15558-1_26
51. Tombari F, Salti S, Di Stefano L (2011) A combined texture-shape descriptor for enhanced 3d feature matching. In: 18th IEEE international conference on image processing (ICIP). pp 809–812. <https://doi.org/10.1109/ICIP.2011.6116679>

52. Unnikrishnan R, Hebert M (2008) Multi-scale interest regions from unorganized point clouds. In: IEEE computer society conference on computer vision and pattern recognition workshops (CVPR workshops). pp 1–8. <https://doi.org/10.1109/CVPRW.2008.4563030>
53. Wang M, Gao Y, Lu K, Rui Y (2013) View-Based Discriminative Probabilistic Modeling for 3D Object Retrieval and Recognition. *IEEE Transactions on Image Processing* 22(4):1395–1407. <https://doi.org/10.1109/TIP.2012.2231088>
54. Werghi N, Berretti S, Del Bimbo A, Pala P (2013) The Mesh-LBP: computing local binary patterns on discrete manifolds. In: IEEE International conference on computer vision workshops (ICCVW). pp 562–569. <https://doi.org/10.1109/ICCVW.2013.78>
55. Xiao J, Hays J, Ehinger K, Oliva A, Torralba A (2010) SUN database: Large-scale scene recognition from abbey to zoo. In: IEEE conference on computer vision and pattern recognition (CVPR). pp 3485–3492. <https://doi.org/10.1109/CVPR.2010.5539970>
56. Zaharescu A, Boyer E, Varanasi K, Horaud R (2009) Surface feature detection and description with applications to mesh matching. In: IEEE conference on computer vision and pattern recognition (CVPR). pp 373–380. <https://doi.org/10.1109/CVPR.2009.5206748>
57. Zeng A, Song S, Niessner M, Fisher M, Xiao J, Funkhouser T (2017) 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In: IEEE conference on computer vision and pattern recognition (CVPR)
58. Zhang Z, Dai Y, Sun J (2020) Deep learning based point cloud registration: an overview. *Virtual Reality & Intelligent Hardware* 2(3):222–246. <https://doi.org/10.1016/j.vrih.2020.05.002>, 3D Visual Processing and Reconstruction Special Issue
59. Zhao G, Pietikainen M (2007) Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(6):915–928. <https://doi.org/10.1109/TPAMI.2007.11110>
60. Zhong Y (2009) Intrinsic shape signatures: A shape descriptor for 3D object recognition. In: 2009 IEEE 12th international conference on computer vision workshops (ICCV workshops). pp 689–696. <https://doi.org/10.1109/ICCVW.2009.5457637>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.