Bridging the Gap Between Local and Global Approaches for 3D Object Recognition

Isma Hadji, Member, IEEE, and G. N. DeSouza, Senior Member, IEEE

Abstract—Approaches to 3D object recognition have dueled over the apparent existence of only two paradigms: local and global. In the first case, both keypoint detectors and descriptors have been defined to capture and match properties of the keypoints themselves or their neighborhoods (also known as their support regions). These approaches proved to be more robust to occlusions and clutter, but have struggled to provide high repeatability in point detection and a general and global enough representation of objects. On the other hand, under the global paradigm, keypoint detectors and descriptors have been unable to handle satisfactorily the same occlusion and clutter more easily handled by their local counterparts. In this research, we bridge the gap between these two approaches. We reveal what we believe is a false dichotomy between the two paradigms by proposing a hybrid approach that leads to better repeatability and better generalization of the representation of objects while increasing the tolerance to occlusion and clutter. First, we propose the Least Expected FeaTure (LEFT) detector, which relies on local features of the objects while these features are selected based on their global and outstanding occurrences within the whole objects. Then, we introduce the Local-to-Global Signature (LGS) descriptor, which takes into account different surface variations across the entire object in order to construct signatures of its local feature vectors. As the experiments demonstrate, the combination of either or both of the two proposed techniques lead to much improved results when compared to SOTA approaches using benchmark datasets. These results support our claim that hybrid solutions between global and local properties are not only possible, but advantageous over each one of them alone.

Index Terms—Object Recognition, 3D, Keypoint Detector, Feature Descriptor.

I. INTRODUCTION

OBJECT recognition is arguably the most important topic in the field of computer vision because of the different applications that this task can serve. Such applications include, scene understanding, robot navigation, tracking, assistive technology and many others.

Any object recognition algorithm consists of three main steps: (i) feature detection, (ii) feature description and (iii) feature matching/classification. These steps constitute the Object Recognition Pipeline. From a computer vision standpoint, the first two steps represent the most important components. In fact, the features used can affect the discrimination power of any classifier used to recognize objects and categorize them.

Over the past decade, several 3D detectors and descriptors have been proposed in the literature. Just as was the case for 2D object recognition, the proposed techniques can be divided into two main categories: namely, local approaches and global ones. Local approaches rely on the description of several small regions of an object using a set of feature vectors. Global approaches on the other hand use a unique feature vector to describe an entire object.

Up until now, 3D Object Recognition has been limited by this false dichotomy between local and global paradigms which has affected both detectors and descriptors. In this work we present ideas that bridge the gap between the two paradigms.

First, we present a 3D keypoint detector that relies on the global structure of objects in detecting outstanding local regions. The proposed keypoint detector was designed with two main goals in mind: (1) Selecting only important and discriminating regions of objects while (2) Increasing keypoints repeatability. To this end, the contributions of this research in terms of keypoint detection are twofold. First, we introduce the concept of finding keypoints considering a global approach, as opposed to more traditional local neighborhood based approaches. Second, we abstract the keypoint selection from experimentally learnt thresholds that depend on low level features for saliency detection. For doing so, we propose the Least Expected Feature criterion (LEFT) for saliency detection. In order to evaluate the performance of the global LEFT detector, the proposed criterion for keypoint detection was implemented with two widely used features; curvatures and shape indices. We compared LEFT detector with three of the most prominent detectors in the literature using those two same features – i.e ISS, KPQ and LSP. As the results demonstrate, our threshold-independent global approach proved to be a much better indicator of point saliency. This lead to high repeatability where the LEFT detector outperformed all other detectors tested in this work on all datasets used.

In addition, we propose a 3D descriptor that further bridges the gap between global and local approaches. While local descriptors proved to be a more attractive choice for object recognition within cluttered scenes, they remain less discriminating exactly due to the limited scope of the local neighborhood. On the other hand, global descriptors can better capture relationships between distant points, but are generally affected by occlusions and clutter. So, we propose the Local-to-Global Signature (LGS) descriptor, which relies on surface point classification together with signature-based features to overcome the drawbacks of both local and global approaches. As the tests demonstrate, the proposed LGS can capture more robustly the exact structure of the objects while remaining robust to clutter and occlusion and avoiding sensitive, low-level features, such as point normals. The tests performed...
on three different datasets demonstrate the robustness of the proposed LGS descriptor when compared to five of the SOTA descriptors today: SHOT, Spin Images, FPFH, VFH and GFPFH.

In general, combining LEFT and LGS lead to a better object recognition performance outperforming all other combinations tested here on the majority of the benchmark datasets used in this work.

The rest of this paper is organized as follows: Section II presents an extensive survey of the different detectors and descriptors used for 3D object recognition. In section III; we describe in details our hybrid approaches and the operation of the proposed algorithms for Keypoint Detection and Feature Description. Section IV: describes in details the experiments, the datasets used and presents all the results obtained. Finally, we conclude with a discussion on the advantages of the proposed algorithms, their limitations as well as possible directions for future work.

II. BACKGROUND AND RELATED WORK

A. Keypoint Detectors

The techniques adopted for keypoint detection in 3D object recognition can be divided into 2 main categories: appearance based and free-form. The former, often, makes use of range images and is inspired by previous methods used for 2D images. On the other hand, free-form object recognition relies on the 3D mesh representation of objects or on the point cloud directly. This research focused on the free-form object recognition.

The most prominent detectors falling in this category were reviewed and compared in [1] using the same datasets used in this work and most recently in [2] using the well known RGB-D dataset. Among these techniques, we find the MeshDoG detector [3]. This method applies the DoG filter to the mesh representation of the 3D object and retains points that represent maxima across different scales. In addition to that, MeshDog retains points that exhibit corner properties by extending the idea of Harris corner detector to 3D. Because MeshDoG applies DoG filter at different scales, it is considered a scale invariant detector. A very similar detector, the Salient Points, is introduced in [4]. It applies the DoG filter directly to the vertices of the mesh creating the effect of the points moving away from their original position. The points whose vertices deviate the most will be the keypoints. Similarly, the 3D detector in [5] extends another 2D detector [6] based on SURF, by building a scale-space out of the voxels, instead of the pixels.

A more dedicated 3D approach for keypoint detection, Intrinsic Shape Signature (ISS) [7], uses the concept of curvatures by calculating scatter matrices for the support region centered at each point of the cloud. By means of eigen-decomposition, the ISS uses the eigenvalues of the scatter matrices to represent the curvature at a point. These eigenvalues are thresholded so as to keep only points with large variation along each direction. A similar approach was presented in [8], and later on referred to as the Keypoint Quality Detector (KPQ) in [1]. Similarly to ISS, KPQ also makes use of the scatter matrices, but instead of using their eigenvalues, KPQ uses the eigenvector in order to calculate a ratio between the two principal axes. Once again, keypoints are selected from those whose ratios surpass a certain threshold. Further pruning is performed in this method by only retaining points that satisfy a certain quality measure based on their tractability. A scale-invariant version of the KPQ is also presented in the same paper, where a characteristic scale is associated with each keypoint by determining at which scale the tractability of the detected points is higher.

Surface curvature is not the only feature employed by detectors in the literature. In Local Surface Patches (LSP) [9], for example, the method relies on the shape indices of the points, while keeping as keypoints those whose shape indices fall far from the mean within a local neighborhood. Finally, the Heat Kernel Signature proposed in [10] also relies on local saliency derived from the maximum heat signature of each point with respect to its neighbors. Thanks to the properties of the heat equation, this detector exhibits high invariance to a wide range of transformations.

As previously mentioned, independently of the type of 3D feature used, all these methods rely on either local, point-wise saliency or fixed thresholds. In this research, we maintain that adoption of a global approach can lead to more robust and efficient detection. Moreover, features that are least expected among the entire point cloud offer more distinctiveness and hence higher repeatability and discrimination potential.

B. Feature Descriptors

As pointed out in Section I, 3D descriptors belong to two main categories: global and local. Local descriptors focus on the local neighborhood of keypoints, while Global descriptors represent the complete object by either using all or a subsample of the detected keypoints. The techniques used to encode the relationships between keypoints can also be divided in two sub-categories: signature-based and histogram-based. A detailed comparison between these two sub-categories can be found in [11].

Among the first successful local 3D descriptors employing histograms we find the Point Feature Histogram (PFH) [12]. PFH describes the angular variations between each two surface normals in the k-neighborhood of a region of interest. Because this descriptor models variations between normals at all pairs of points, it is very computationally intensive. For that reason, Fast PFH (FPFH) was introduced in [13] to speed up the process, but in detriment to its power to discriminate objects. An alternative was then introduced in [14], where the authors proposed a 3D extension of the 2D Shape Context descriptor [15]. This descriptor relies on the position of keypoints in a local neighborhood (support region) with respect to a virtual spherical grid. Each spherical grid accumulates a weighted sum of the points falling in that grid. In order to achieve repeatability, the north pole of the spherical grid is first aligned with the direction of the surface normal at the point being described. This alignment is sensitive to the choice of the reference frame – a major issue for any descriptor based on surface normals. So, a major breakthrough in terms of 3D
descriptors was introduced by the SHOT descriptor ([11]), where the authors addressed this problem with the reference frame repeatability by attaching a unique reference frame to each point. Once the reference frame is determined, the local support of each point is discretized in a way similar to that in the 3D Shape Context descriptor. Another contribution of SHOT was that the histogram of each grid is concatenated to form the final signature. This technique for defining local repeatable reference frames was later used to improve the 3D Shape Context in [16] by taking the initial representation proposed for 3D Shape Context and disambiguate it using the unique reference frame introduced by the SHOT descriptor. Although quite different in terms of their implementations, the afore mentioned techniques are very similar in concept. Their reliance on local neighborhoods and low-level features, such as surface normals, compromise their ability to discriminate objects and make them quite sensitive to viewing perspectives and noise.

As far as signature-based descriptors are concerned, Point Signatures (PS) [17] is possibly the best known descriptor. It relies on point positions also within a local neighborhood, but it encodes these positions in the local sphere in terms of the angle between the surface normal at the point and the signed distance of the points to the plane separating the sphere into two halves. While this method captures exactly the structure of the local neighborhood, it remains very sensitive to small changes in the normal estimation and the reference frame. Another signature-based descriptor is proposed in [8], where signatures are built from the depth values of the local surface after its normal vector has been aligned with the Z axis. Also, in order to reduce the dimensionality of the signature, the method resorts to a PCA subspace of the feature vector. Although signatures usually lead to a better discriminating power, the reliance on local neighborhoods causes low repeatability in keypoint matching since many keypoints in one object can have similar local structures.

Finally, when it comes to global descriptors, the View-point Feature Histogram (VFH) [18], which is an extension to the FPFH, is one of the most widely used. Instead of describing relationships between points in local neighborhoods, VFH does so for every point in the cloud with respect to their centroid. In addition to this, VFH introduces view point variance – an important addition when it comes to estimating object pose. Another attempt of generalizing FPFH was provided in Global FPFH [19]. In this case, FPFH descriptors are used locally for every point in the object and a conditional random field is trained in order to classify the collected local descriptors into a set of primitives. Next, the relationships between keypoints are encoded by counting the number and type of transitions between primitives while traversing keypoints in an octree. Later, in Global Radius-based Surface Descriptor, GRSD ([20]), a modification to GFPFH was proposed by eliminating the first classifier and adding a geometric solution to the shape primitives. A similar approach called Global Structure Histogram, or GSH, was also proposed in [21] with the goal of capturing the global structure of the object. In order to do so, the authors followed the same procedure proposed in [19], but using a clustering algorithms to learn the points classes and by encoding the relationships between keypoints in all clusters using geodesic distances.

In this work, we maintain that adopting the strengths of both local and global descriptors can lead to highly discriminating features. These features can be successfully used in the task of object recognition and pose estimation within scenes, even if they suffer from clutter and occlusion.

III. PROPOSED METHODS

The goal of this work is to merge ideas from the local and global paradigms in order to achieve a more robust recognition. Specifically, in this work we focus on the two most important steps involved in object recognition – i.e. Keypoint Detection and Feature Description.

A. Keypoint Detection

Object recognition algorithms rely on the detection of a subset of important or discriminative visual stimuli (keypoints). In general, keypoint detection is based on two main steps: detection and pruning. First, a criterion for detecting local saliences is established. One example, is to select as keypoints those points whose curvature within a small neighborhood is above a user set threshold. Once the keypoints are detected, another pruning step is usually used in order to keep only most relevant keypoints, such as keeping points with local maxima [22].

Independently of the type of 3D feature used, all 3D detectors rely on a local criterion during the detection step. This criterion is usually a point-wise salience measure that is based on experimentally learnt thresholds. In this research, we question both the threshold based approach, as well as the local characteristic of traditional 3D keypoint detection schemes, and we evaluate the effect and relevance of each one of them.

First, in this research we maintain that using learnt thresholds based on low level feature values is very detrimental to the repeatability of the keypoints, because of the high sensitivity of these thresholds to density variations and noise levels. For that reason, in our implementation we abstract the feature selection from the actual feature values or thresholds. While doing so, we propose to select keypoints based on the distribution of low level features. We dub this technique Least Expected Feature (LEFT). The rationale of LEFT is that least frequent features are more likely to uniquely represent a region of interest in the object. This rational can be applied to any type of low level features under different scenarios.

In a first case study, we evaluate the detection criterion itself and compare it with traditional approaches that detect keypoints in a local neighborhood based on experimentally learnt thresholds. To this end, we adopt the LEFT criterion to the local neighborhood.

In a second study, we question the local characteristics of keypoint detectors and we propose the adoption of a global approach towards a more robust and efficient detection – i.e. a global approach for selecting keypoints based on the distribution of the features in the entire cloud of point as opposed to local neighborhoods. In fact, when it comes to
describing objects, global descriptors such as VFH [18] have been widely accepted. However, to the best of our knowledge, a global approach for 3D keypoint selection has not been explored yet.

1) Motivation: We are convinced that a global view for keypoint detection offers several advantages over the traditional local (point-wise) approaches. First, it ensures that non-interesting points that exhibit high local variations are ignored. In fact, while traditional 3D keypoint detectors focus on finding keypoints in the local neighborhood of each point in the objects, the proposed method looks beyond the local neighborhood and relies on the entire structure of the objects in detecting outstanding regions. This makes the keypoints detected by the proposed method more representative of the object to be described and hence recognition of those same objects can be facilitated. To illustrate this idea, we extracted keypoints from three sample household objects, using both the ISS detector as well as the proposed Global LEFT detector, relying in both cases on points curvatures as the underlying low level features.

As Figure 1 clearly shows, ISS detector selects keypoints in different parts of the objects most of which are neither characteristic of the object nor relevant to the following description stage. In fact, a good keypoint should represent a region of the object that contains enough information to uniquely characterize the point and hence the object to which it belongs. Therefore, points on the body of the bottle and the spray or the edges of the pan are not good keypoints since they are common to many objects. For example, keypoints on the body of the bottle and the edge of the pan can cause confusion between those two objects. Let alone that points on edges are potentially unstable and therefore less repeatable. On the other hand, because the proposed detector looks at the global structure of the objects, it only selected characteristics parts of the objects as keypoints –i.e. the handles of the pan, the nozzle of the spray and the cap part of the bottle. This ensures that descriptors extracted later on using those keypoints are specific to the object and thus facilitate accurate recognition.

In addition to that, the proposed algorithm allows for arbitrary percentages of the cloud of points to be selected as keypoints, which can be useful to accommodate for noise levels of the sensor and other application dependent requirements. Finally, since LEFT is a criterion that can be applied to any 3D feature, it opens the possibilities to a large number of different detectors by simply changing the underlying feature. The details of the LEFT algorithm are presented in the following.

2) Least Expected Features Detector – LEFT: The proposed keypoint detection scheme was inspired on the way animals recognize “important information”, whether it is for attention or gaze in avoiding threats [23]. Intuitively, it is easy to accept that the least frequent a specific shape, the more representative it is of the object. In that sense, we propose a series of detectors based on the Least Expected Features criterion (LEFT). Unlike other detectors, the LEFT detectors select features not based on a local criterion or a threshold, but rather on their frequency of occurrence in the entire object. As mentioned in Section II, curvature and shape index are well accepted features for 3D detectors. So, the LEFT-based detectors used in this research include: Least Expected Curvatures (LEFT-C), Least Expected Shape Index (LEFT-SI).

Basically, the general algorithm for detecting keypoints using LEFT is built around 4 main steps:

1. Low level features describing a 3D cloud of points or mesh are estimated for each point $p_i$ in the object.
2. Most similar features are grouped together, therefore quantizing the feature space into histograms. In other words, points having similar feature values are assigned to the same bin of a histogram. For example, for our LEFT-C detector, we quantize the curvature values into a d-dimensional histogram whose upper and lower bound correspond to the minimum and maximum curvature values respectively. All points whose curvature $c_i$ is $c_k \leq c_i \leq c_{k+1}$ will be added to bin $k$. Therefore, the built histogram maintains a count of the frequency of occurrence of the feature values present in the object.
3. The histogram built in the previous step is sorted in ascending order. In other words, the features are sorted from least to most frequent.
4. Points with least frequent feature occurrence in the object are selected as keypoints starting from the points in the first bin and adding more points until the desired number of keypoints is reached. Therefore, the size of the histogram does not play a major role in the algorithm since keypoints are chosen from those falling in the first few bins, going from the first bin to the next one until the desired number of keypoints has been selected. The user is given the option to select the desired number of keypoints according to his application. In our implementation we set the size of the histogram to 100 bins.

We propose to select keypoints globally as opposed to all
other methods proposed in the literature that solely rely on the local neighborhood. It turns out that adopting a global approach allows for the selection of very distinctive keypoints across the entire object. An example of the keypoints detected by the Global LEFT-C detector is shown in figure 2(a) and compared to keypoints detected using the same feature but using the ISS criteria introduced in [7]. From the Figure it is clear that our criteria only selects outstanding points as opposed to the ISS detector that selects points all over the object even in smooth non-important regions on the body of the dragon. Therefore, one would expect that descriptors of keypoints on smooth regions of the dragon would be mostly similar and not discriminating as opposed to the distinctive keypoints detected by the proposed Global LEFT-C.

It is worth noting here that while our global approach yields very distinctive keypoints, it may require a pre-processing segmentation step for detecting keypoints on objects within scenes. However, this is not a disadvantage of our method, given that the usual global object recognition pipeline requires the concept of objects and therefore it involves scene segmentation as a pre-processing step by default as argued in [24].

In the next two subsections, we will describe in greater detail the low-level features used in this research. These low-level features are also the ones used in the approaches used for comparison with our detectors – namely: ISS and KPQ that use curvature values and LSP which uses shape index values.

Curvatures: Many methods have been developed to estimate surface curvatures on unorganized cloud of points as described in [25] and [26]. These methods can be divided into two main approaches. The first approach relies on fitting the cloud of point into a surface and extracting the principal curvatures of the fitted surface using Minimum Least Square estimation. The second approach, adopted in this work for its speed, is to describe each point’s curvature using the eigenvalues of the covariance matrix constructed from the point’s support region (i.e. the points in a sphere of radius \( r \) around it). Assuming \( \lambda_0 < \lambda_1 < \lambda_2 \) are the eigenvalues of the covariance matrix, One estimate of the curvature of a point \( \sigma_n(p) \) is how much it deviates from the tangent plane with respect to the total variation.

\[
\sigma_n(p) = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2} \tag{1}
\]

In the case of the Least Expected Curvatures (LEFT-C), we first group points with similar curvature values throughout the whole object. Then, we retain the least frequent of those points.

Shape Index: Another well accepted low level feature used by many detectors is the Shape Index (SI). The shape index was first introduced by Dorai and Jain in [27] as a way to associate an estimate of the shape of a surface with each point, and specifically to measure how concave or convex a surface is. The shape index is defined by eq. (2) for the two principal curvatures, where \( k_1 \) and \( k_2 \) represent the maximum and minimum principal curvatures respectively. Ever since, many proposals for 3D keypoint detectors, such as [7], [22], have used the SI as a base feature for their detector.

\[
S_i(p) = \frac{1}{2} - \frac{1}{\Pi} tan^{-1} \frac{k_1(p) + k_2(p)}{k_1(p) - k_2(p)} \tag{2}
\]

We use the shape index as one of the features under the LEFT criterion. We call this method the Least Expected Shape Index (LEFT-SI). It follows the same idea described for LEFT-C but using SI as our base feature.

B. Feature Description

Designing good feature vectors, or descriptors, is the most critical step involved in 3D object recognition. Indeed, the descriptors have the greatest effect on the overall recognition result, as argued in [24].

As mentioned earlier, the techniques adopted for object description can be divided into two main categories; global or local. We refer the reader to Section II for a detailed comparison between the two types. At first, global descriptors seem a more natural choice, since they seek to encode the entire set of keypoints into a single feature vector describing the object. This makes global features far more discriminating given that the entire geometry of the object is taken into account. However, being able to observe as much as possible
Recently, global 3D descriptors introduced interesting ideas involving the classification of object surfaces based on their shapes. These ideas should ultimately increase the discriminating power of the descriptor. However, it is clear that a great amount of information is lost in the process of encoding the relationships between all keypoints of an object into a single feature vector. In addition to that, global descriptors are very sensitive to clutter and they require a good segmentation algorithm as a pre-processing step ([24]). Also, even if a perfect segmentation could be obtained, occlusions can have a major effect on the performance of global descriptors. On the other hand, a common problem that may affect all local descriptors is the inability to discriminate similar support regions not only within the same object but also across different objects. To illustrate this idea, we extracted both the SHOT and the LGS descriptors from two different keypoints coming from two completely different objects. We set the radius size of the support region used to estimate the SHOT descriptors to 25 times the mesh resolution – i.e. almost twice the recommended size of 15 times the mesh resolution [11]. As Figure 4 clearly shows, the SHOT descriptors for these two keypoints are almost identical even though they come from very different objects, while the LGS makes clear distinction between the same two keypoints.

For these reasons, the proposed descriptor starts from a local approach, where each keypoint is represented with a unique feature vector – i.e descriptor. Then, it turns global by looking at the entire object while capturing the structure of the object with respect to that keypoint.

2) Local-to-Global Signature Descriptor – LGS: As we just mentioned, the main advantage of the LGS descriptor resides in the idea of looking beyond the local support.

1) Motivation: The proposed LGS descriptor can be regarded as a bridge between local and global paradigms. In this research we propose a new descriptor built at the up-to-now unseemly intersection between these two paradigms. The Local-to-Global Signature – or LGS descriptor – is local in the sense that each feature vector describes a single keypoint, rather than the entire object, but at the same time global as it looks beyond the local neighborhood (support regions) to describe the properties of keypoints with respect to the entire object. Unlike traditional global descriptors, the LGS overcomes issues related to occlusion and clutter by constructing signature vectors instead of histogram-based vectors. The advantages of using signatures to reduce the effects of occlusion are further detailed in Section III-B2, but again, that is not the only contribution of the proposed descriptor. In summary, the LGS was built around the following five main ideas: 1) Relying on surface point classification to capture the entire geometry of the object (global property); 2) Describing keypoints (local property), but using both local and global support regions grouped by the same surface class; 3) Using signatures (global property) to avoid loss of information and mitigate the effects of occlusion; 4) Using distributions of L2-distances to encode the relationship between keypoint and support regions to increase robustness to noise and eliminate the use of sensitive features such as surface normals; and 5) Using confidence on the relationship above to improve the matching during object recognition.

Figure 3 illustrates the overall construction of the LGS descriptor. These ideas will be further detailed in the remaining of this section.

Figure 3. Steps of the construction of the LGS descriptor: yellow dots represent keypoints to be described; points are color coded according to the surface class to which they belong, from sharp regions in red to smooth surfaces in light blue; the signature consists of the distances f between the keypoint and its local and global support regions, plus the corresponding confidences in their assignments.
In fact, it looks at different regions representing different properties of the object. This is accomplished by: (i) assigning classes to all the points in the cloud (global property); (ii) describing keypoints (local property), but using both local and global support regions; (iii) using signatures to describe the relationships between keypoints and selected points from all the assigned classes; (iv) using L2 norm to robustly encode such relationships; and (v) using confidence on the relationship above during matching.

**Point Classification** : To classify the points we use the radius-based surface classification proposed in the RSD descriptor ([20]). Our technique starts form the assumption that every two points fall on a sphere. Therefore, within each neighborhood the radii of all virtual spheres are estimated using points locations. Then, the maximum and minimum sphere radii present in the local neighborhood are derived. For more details we refer the reader to [20].

In the original proposal of RSD, the authors used both the minimum and maximum radii to classify points as belonging to one of the geometric primitive shapes. In our implementation however, we chose to classify points from very sharp to very smooth. Our decision for this classification approach is motivated by one main argument: to be able to find a pre-defined number of classes independently of the object considered. This may not be feasible when using shape primitives where some specific shapes may not appear in all objects. In addition, since this classification scheme is independent of primitive shapes, it allows the algorithm to vary the number of classes by simply varying the ranges of sharpness and smoothness. For example, we cannot expect to find a toroid shape in all objects, but we can reasonably assign different levels of sharpness or smoothness to any surface based on its curvature.

In our implementation, we rely only on the minimum radius from the RSD method described above, where a very small radius is an indication of a very sharp surface, and a large radius represents a smooth one. After deriving the minimal radii associated with each point on the surface of the object, the algorithm can split the radii values into \( N \) different ranges, representing the \( N \) classes of the object’s surfaces. If for example \( N = 3 \), any point with radius \( r < \alpha \times \text{mesh resolution} \) would be assigned to class 1; a point with radius \( \alpha \times \text{mesh resolution} \leq r < (\alpha + 5) \times \text{mesh resolution} \) is assigned to class 2; and all other points are assigned to class 3. In our experiments, the value of \( \alpha \) was empirically set to 6.

Since LGS uses continuous ranges of radii values for surface point classification, fuzzy regions may emerge. These fuzzy regions contain points with values of radii that are close to the end of one range and the beginning of the next one. Therefore, these points could belong to any of the two consecutive ranges. It is easy to understand that these regions are unstable and points can move from one class to another as noise is added or removed. Also, noise can cause spikes to appear on otherwise smooth surfaces. Therefore, to cope with these potential instability in surface classification, we propose the assignment of both a class and a membership to the class for each point in the cloud. Basically, after the initial crisp classes are assigned to each point, the algorithm searches over each point and calculates a coefficient of confidence that the specific point belongs to the assigned class. These confidences \( c \) approximate the probability that a current point \( p \) belongs to class \( n \) given the number of points from that class in its local support region; that is:

\[
c = \frac{\# \text{ of points in class } n \text{ in the neighborhood of } p}{\text{Total number of points in the neighborhood of } p}
\]

The rational behind these confidences is the assumption that in any small neighborhood points are more likely to belong to the same surface class. Also, a low confidence indicates that the point belongs to a fuzzy or noisy region, where multiple classes may be assigned. This determines when the algorithm is to give high or low weights to the points used in the next steps of the algorithm.

**Signature Construction** : Once all points are assigned to one of the \( N \) classes and their confidences are calculated, the LGS descriptor is constructed in a signature-based fashion. Figure 3 should help the reader in understanding the following steps. First, for every keypoint, the algorithm finds its corresponding k-nearest neighbors within each class. Next, the k-neighbors in each class are sorted based on their distance from the keypoint and divided into \( D \) clusters. Then, the median L2-distances from the keypoint to the points falling in each cluster form a \( D \)-dimensional feature vector. These distances are the actual features \( f_l \) of the LGS descriptor. Finally, feature vectors representing the neighborhoods of the keypoint in each one of the \( N \) classes are concatenated to form the final signature whose length is equal to \( D \times N \). Again, Figure 3 illustrates the construction of a simplified signature in the LGS descriptor.

It is important to mention again that the main motivation for using a signature-based descriptor is its potential robustness to occlusions. In fact, if parts of the object happen to be occluded in the scene, only the entries corresponding to those parts of the object will be altered in the LGS descriptor, while the rest is unchanged. On the other hand, a histogram-based descriptor would be completely affected if the support region of a keypoint is partially occluded.

**Descriptor Matching** : In parallel to constructing the LGS descriptor, the algorithm builds a second feature vector filled with the confidences \( c_l \) corresponding to each feature point \( f_l \). Once again, this idea is illustrated in Figure 3 for a very simplified case. These confidences are used as weights during the matching stage, when the LGS computes the distance \( d_{ij} \) between a pair of signatures \((i, j)\). In other words, the distance between each entry of the pair of signatures is multiplied by the corresponding minimum confidence. This allows LGS to reduce the effect of unstable points located on fuzzy regions as discussed in the beginning of this section. Mathematically, the weighted distance used to compare LGS descriptors is given by:

\[
d_{ij} = \sqrt{\sum_{l=1}^{(D \times N)} \min(c_{li}, c_{lj}) \times (f_{li} - f_{lj})^2}
\]

where \( c_{li} \) and \( c_{lj} \) are the confidences of points in the pair \((i, j)\), \( f_{li} \) and \( f_{lj} \) are their actual features, and \( D \) is the dimension of the feature vector.
IV. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we validate the proposed algorithms by undertaking a thorough experimental evaluation of their capabilities both independently and when combined. We demonstrate their advantages and contributions in different challenging datasets, and we justify any choice of parameters by investigating their effect in the solutions obtained. We used three well-recognition datasets from the literature [1], [11]:

1) The first dataset consists of synthetic data with added noise at three levels equal to 10%, 30% and 50% of the mesh resolution. It is referred to in [1] as Retrieval dataset where both scenes and objects are full 3D models.

2) The second dataset is known as the Random Views dataset. In this dataset, the models are full 3D models, while the scenes are 2.5D views. We chose this dataset because of the added challenge where a full 3D model is matched to one view in the scene.

3) The third dataset is a real datasets acquired using stereo cameras. Obviously, given that it is a real dataset, it presents the biggest challenge for any object recognition algorithm due to: the level of occlusion and clutter; the presence of smoother and more similar objects, and the larger reconstruction errors between scenes and models.

Figure 5 presents examples of models and scenes from the dataset used.

As it was done in [1], in all our tests, a keypoint extracted from a model object $k^m_i$ is considered repeatable if we can detect the point $k^s_i$ again in a scene, where the model is rotated and translated, in the same position within a minimal error threshold $\epsilon$. In all experiments we set a strict error threshold that is equal to two times the mesh resolution.

$$||R_{ms}k^m_i + T_{ms} - k^s_i|| < \epsilon$$ (5)

The steps followed for measuring repeatability are as follows:

1) For each pair of models and scenes find the set of keypoints $k^m_i$ and $k^s_i$, respectively.
2) Rotate and translate the model based on the ground truth rotation $R_{ms}$ and translation $T_{ms}$.
3) Find correspondences between detected keypoints in the model and in the scene.
4) Calculate the relative repeatability as the number of correspondences divided by the total number of unoccluded keypoints detected in the model.

We should note here that in order to be fair to other detectors, we introduced one additional step above for the Kinect dataset only. This dataset contained ground and wall planes. However, since our global detector relies on detecting keypoints with least frequency, one could argue that due to the large number of points on such planes, most of the selected keypoints would lie on the objects themselves. That would ultimately increase the repeatability of our detectors. So, for this reason, we removed the planes from the Kinect scenes.

Also, for our global approach, one could argue that since our detectors rely on the distribution of the features in the entire cloud, their performance for full scenes in the presence of multiple objects would degrade due to the effects of those objects on the same distribution. For example, if our scene contains a model with a relatively similar shape, it might happen that no point is detected on that model. Figure 6 illustrates such a case, where a sphere that has evenly distributed curvatures is placed with other objects in the scene. In that case, virtually, no keypoints are detected on the sphere in the scene. For this reason, for our global keypoint detector, we consider an approach where each scene is segmented first and the distribution of features is derived for each segment separately. This way, the results of the distribution are less affected by the presence of other objects. Again here, this is a required pre-processing step for any global object recognition pipeline and it is therefore fair to use it with our approach since we are proposing a global keypoint detector.

As mentioned in the introduction, in this research we both question the threshold based techniques for keypoint selection as well as the reliance on local neighborhoods of traditional approaches. For this reason, in our experiments we first apply our LEFT criterion to the local neighborhood – i.e. we select keypoint based on their frequency of occurrence within the local neighborhood to highlight the downsides of using experimentally learnt thresholds. We refer to this in our tests as Local LEFT. Then, we evaluate the effect of the local neighborhood in the keypoint selection by applying the same criterion to the entire object.
All tested detectors also depend on the scale, or size, of the keypoints detected. This feature used also has an important role. In fact, all tested detectors were introduced in [30].

A simple region growing algorithm based on Euclidean distance was employed to set, no matter the different combinations tested for radii and thresholds. However, in all cases, the different thresholds for ISS, KPQ and LSP have been tweaked so that a comparable number of keypoints is selected across the different detectors used.

In this test the only fixed value was the scale used for features estimation that we set to 6 times the mesh resolution. Also, in all experiments we fixed to 100 the number of bins for the histograms used in quantizing the feature space for the proposed LEFT detector. Figure 7 summarizes the results obtained for all 4 datasets. The number of points selected is reported in terms of average percentage of keypoints selected from the entire object.

As this experiment clearly demonstrates, for both the local and global approach, our LEFT criterion results in higher repeatability rates. This is thanks to the fact that LEFT abstracts the keypoint detection from specific thresholds. Therefore, it only selected outstanding regions in the entire objects and resulted more stable keypoints.

We tested our global LEFT detectors both with and without segmentation. It is clear from figure 7 that our global approach, together with a good segmentation, results in much higher repeatability rates. As far as the segmentation is concerned, we used a simple region growing algorithm based on Euclidean distance as was introduced in [30].

In addition to that, these results suggest that the low level feature used also has an important role. In fact, all tested detectors based on shape index have a worst behavior. This is also an indication of the importance of the quality of the keypoints detected.

2) Repeatability of Detectors for Variable Support Size: All tested detectors also depend on the scale, or size, of the neighborhood \( r \) chosen to estimate the corresponding low level features. For that reason, in this experiment we tested the repeatability of the proposed detectors for varying support sizes. Since the different datasets employed different sensors, with different resolutions, we standardize the measurement of the scale as a function of the instrument resolution. That is, the results reported in this paper for different datasets are shown in terms of mesh resolution. This approach was also employed by [1]. Based on the results in section IV-A1, for this experiment, we select the thresholds that yielded the best repeatability for the detectors used for comparison and we use the corresponding percentage of keypoints for our method. Yet, we try to keep the number of selected keypoints by all detectors to a similar number on average. For example, for the LSP detector we didn’t select the thresholds that gave the best repeatability since that accounts to selecting around 15% of the cloud as keypoints, which is almost 3 times the average number of points selected by the other detectors.

The results obtained with the different detectors are reported in Figure 8. Interestingly enough, these results suggest that the scale does not play a major role in the keypoint selection. In fact, the only dataset where larger scales involved a small drop in the repeatability is the Random views dataset. In this dataset 3D models are matched to 2.5D scenes. Therefore, larger scales implies relying on occluded regions of scenes when estimating features and detecting keypoints, yielding therefore different feature values between model and scenes and thus different keypoints are selected. In all other cases, the change in scale affected the keypoint detection equally in both scenes and models and hence did not affect much the repeatability of the keypoints.

B. Descriptor Matching Experiments

In order to evaluate the discriminating power of the proposed LGS signature, we devised testing scenarios that highlight the robustness of LGS for the case of a model-scene matching framework.

In that sense, we present two sets of experiments: The first experiments are mainly concerned with the parameter selection, while the second experiments present a quantitative comparison with other 3D descriptors proposed in the literature. In particular, we compared the LGS descriptor against three of state of the art descriptors: i) FPFH ii) Spin Images and iii) SHOT. We chose FPFH and Spin Images as representatives of histogram-based local 3D descriptor and also because they are arguably the most widely used descriptors in the field. Also, we selected SHOT for comparison with a descriptor based on a signature of histograms, and again because it represents one

Figure 6. Detected Keypoints using LEFT-C on a synthetic dataset. (a) Without segmenting the scene no keypoints are detected in the models with similar curvature such as the half sphere. (b) The detected keypoints using LEFT-C on the half sphere model. (c) The detected keypoints in the segmented scene. Note how, this time keypoints are detected on every object in the scene.
of the state-of-the-art local 3D descriptors, achieving the best results on the datasets used here.

C. LGS Parameter Selection

As explained in Section III-B2, the proposed LGS descriptor may be affected by the choice of two parameters: the number of classes \( N \) used to construct the signature and the number \( k \) of neighbors used in each class. In this section, we present an experiment using two different datasets highlighting the effect of each one of these parameters on the performance of the LGS descriptor. This also provides the reader with an intuition regarding how they can be set.

In general, a descriptor \( T \) is considered robust if for any given keypoint in the scene, its exact correspondence can be found in the model. The steps followed for establishing correspondences are as follows:

1) For each model-scene pair \((m, s)\), find the set of keypoints \( k_m \) and \( k_s \) and describe each keypoint using the \( T \) descriptor.

2) Compare the descriptor for each keypoint in the scene against all the descriptors in the model and take as a correspondence the descriptor from the model with the shortest distance – for the LGS, that is given by eq (4).

3) Find the true correspondences using the provided ground-truth transformation between model and scene.

4) Compare the correspondences from step 2 and step 3 and count total number of true/false matches.

In order to abstract the performance of the descriptor from the effect of non-repeatable keypoints, we followed the same framework proposed in [11] for keypoint detection. More specifically, we down-sampled each model so that only 5% of the original cloud was kept. These will be the model keypoints. Then, using the ground-truth transformation, we found the corresponding keypoints in the scene. These steps guaranteed 100% repeatable keypoints.

We opted for down-sampling the keypoints to ensure that these are uniformly distributed over the object and that the algorithm is not biased by the type or location of the keypoints.

Figure 7. Repeatability versus the number of keypoints for Retrieval dataset with (a) noise =0.1, (b) noise=0.3, (c) noise=0.5. Random Views dataset with (d) noise =0.1,(e) noise =0.3, (f) noise =0.5, (g) Stereo dataset.
Number of Classes: As previously mentioned the first step towards building the LGS signature is to classify points based on the surface type to which they belong. In particular, we classify surfaces from very sharp to very smooth. In this experiment, we highlight the effect of the number of classes by varying them from 1 to 5 classes, and allowing accordingly from 1 to 5 shades of sharpness/smoothness. Figure 9 summarizes the results obtained for variable number of classes when the number of neighbors is fixed at 300 nearest neighbors in each class. It is worth noting that for a number of classes equal to 1, the LGS descriptor approaches a local signature-based descriptor where all nearest neighbors are close to the described keypoint.

As we can see from Figure 9, the number of classes plays an important role in the performance of the LGS. This becomes particularly clear for the Retrieval dataset, where changing from two to three classes improves the percentage of good correspondences by as much as 15% (at noise level equal to 0.5). This dataset contains objects with highly different surfaces and shapes, therefore a small number of classes is not discriminating enough. As one would expect, a small number of classes does not capture enough of the variations in the structure of these objects. On the other hand, adding...
too many classes also leads to reduced discriminating power. We attribute this to the drop in the classification accuracy that affects the regions used in building the signature. In fact, increasing the number of classes involves using smaller ranges to assign points to different surface types. Therefore, forcing the presence of more fuzzy and unstable regions whose points can be assigned to any one of the neighboring ranges.

In order to support this claim, we performed an evaluation of the classification accuracy versus the number of classes. For each point in the model, we let the algorithm find its corresponding point in the scene and we checked whether they had been assigned to the same class (see Figure 10). As we claimed, by adding more classes we observed a drop in the classification accuracy, which ultimately caused a drop in the number of good correspondences.

**K-Neighborhoods:** The LGS descriptor consists of concatenating features from the $k$-closest neighbors in each class. So, the number of neighbors does play a role in the stability of the signature. In fact, a very small neighborhood implies relying more heavily on points that fall close to fuzzy regions – i.e. junction regions between two classes. As mentioned earlier in Section III-B2, these are non-stable regions since points on those regions can switch classes very easily in the presence of sensor noise. On the other hand, neighborhoods too large can cause the algorithm to look beyond stable regions, likely falling on the next fuzzy region. In addition to that, very large neighborhoods may imply using much bigger parts of the object which can therefore cause the LGS signature to become more affected by occlusions.

Once again we validate these claims by evaluating the effect of different neighborhood sizes on the discrimination of the LGS descriptor. Given the results obtained in the previous experiment, we fixed the number of classes to 3 for this test. We varied the neighborhood sizes from 100- to 1000-nearest neighbors per class. It should be noted here that given the definition of the LGS signatures, varying the neighborhoods sizes implies varying the length of the signatures. In particular, as previously mentioned, in the algorithm for the LGS, each neighborhood is split into smaller clusters and the median distance of each small cluster is used. For example, in our implementation, we used clusters with 10 points each, therefore for a number of classes $N = 3$ and a neighborhood size $k = 100$, we have the number of clusters in each class $D = 100/10$, leading to a signature with dimension $D \times N = 30$.

Figure 11 summarizes the results obtained from varying the neighborhoods sizes. These results prove two main points: (i) we can see that the percentage of good correspondences increases in the beginning as we use larger neighborhoods. Again here, this is more evident with the Retrieval dataset where increasing the neighborhood size from 100 to 300 per class improves the percentage of correspondences by about 20% in some cases (at noise level equal to 0.5). This confirms our initial statement regarding the importance of looking beyond the local neighborhood when constructing local descriptors; (ii) interestingly enough, we can see that the LGS descriptor is not too sensitive to the use of larger neighborhoods. We attribute this to the advantages brought by using a signature-based feature vectors. This could be seen as making the LGS less sensitive to occlusions.

**D. LGS Matching Capability**

In this experiment, we validate our proposal in terms of matching capability, using the Recall versus $1 - Precision$ metric that captures both true and false positives as argued in [11], [31], [32]. In order to find potential correspondences, for every feature in the scene, the algorithm computes the first and second nearest neighbors in the model. Then, a match is established between the scene feature vector and its nearest neighbor in the model if the ratio between the first two nearest neighbors is below a certain threshold, as suggested in [11], [28]. This threshold is the value that is varied from 0 to 1 in order to produce the Recall versus $1 - Precision$ curves in Figure 12. A correct match is counted as a true positive, and as a false positive otherwise. The total number of correspondences is known from the ground truth and therefore **Recall and $1 - Precision$** are calculated as follows:

$$\text{Recall} = \frac{\text{True Positives}}{\text{Total number of correspondences}} \quad (6)$$

$$1 - \text{Precision} = \frac{\text{False positives}}{\text{False positives} + \text{True Positives}} \quad (7)$$
As far the parameters used in this experiment, for the LGS descriptor, we picked the best parameters learnt from the previous experiment. In particular, we use number of classes $N = 3$ with the number of neighbors per class $k$ set to 300. Given that we split each neighborhood to clusters of 10 points, this lead to a 90-dimensional signature. For the other descriptors used in this comparison, the main parameter to be set is the radius size of the local support region of the keypoints. In this case, we used the same size recommended in [11] – i.e. 15 times the mesh resolution.

As the results demonstrate, in most cases the LGS proved to be more discriminating than the different local descriptors tested in this paper. This should support the claim regarding the importance of looking beyond the local neighborhood for keypoint description. In addition, the use of signature proved to hold a higher discriminating power since LGS and SHOT always outperformed the histogram-based approaches. Also, given that the LGS descriptor does not directly rely on sensitive features, such as point normals, it was less affected by noise as Figures 12 (a) through (f) demonstrate (we discuss case (g), next), this is also clear on the Random Views dataset when comparing the behavior of LGS and SHOT for increasing noise levels as opposed to Spin Images for example whose performance was drastically affected by the addition of noise. Also, it is worth noting here the level of difficulty involved when matching full 3D models to partial views where the Random Views dataset turned out to be the most challenging resulting in sub-optimal results with all tested descriptors.

One limitation of using the proposed approach is highlighted in the last experiment using the Stereo Dataset. In this case, the results for the LGS where only better than the Spin Images. In fact, as previously mentioned in Section IV-C, for the Stereo dataset, both models and scenes are reconstructed from different stereo pairs, which causes differences in the shapes of the objects found in the scene and the model. This in turn leads to a less accurate surface classification (Figure 10), which ultimately affects the LGS signature. In fact, many of the points in the model are assigned different classes from the ones in the scene. As a consequence, the support regions
found by the algorithm in each class of the model are also different from the ones found in the scene.

In order to further illustrate the effect of this severe misclassification observed on this last dataset, we simulated 100% good classification of the points. Specifically, we first classified points in the models into 3 classes as described in Section III-B2. Then, using the ground-truth transformation, we assigned the same class to the corresponding points in the scenes. Finally, using this simulated classification results, we constructed the LGS descriptor as usual. As can be seen from Figure 12(g), provided a good classification, the LGS signature can still achieve higher discrimination than any of the tested descriptors. While this can be seen as a limitation of our method and indeed requires future improvements, it also further proves the benefits of looking beyond the local neighborhood and encoding more of the structure of objects in describing each keypoint. In addition, the four experiments together prove that relying on a signature instead of a histogram allows LGS to capture more details, increasing therefore the discriminating capability of the descriptor.

V. COMBINING LEFT AND LGS

The ultimate goal of the developed algorithms is to achieve accurate object recognition given a good detector/descriptor pair. While, we focused on the performance of the proposed LEFT keypoints detector and LGS feature descriptor separately in Sections IV-A and IV-B respectively, the rest of the experiments will focus on the combination of the two and their role in achieving higher recognition rates.

A. LEFT-LGS Matching Capability

Before plugging the proposed LEFT detector and LGS descriptor in the Object Recognition Pipeline, it is interesting to study the effect of combining detectors and descriptors on the descriptors matching capability itself. For this reason, we decided to reproduce experiment IV-D however in this case we take into account the effect of the detector on the number of correct and false matches. While in Experiment IV-D we were abstracting the results of the descriptors from the performance of the detector by simulating 100% repeatable keypoints, in this case we use the proposed LEFT keypoint detector in the first stage of the experiment. Also, in order to further investigate the effect of using the LEFT detector on the descriptor matching, we decided to perform the same experiments using ISS detector for comparison purposes. We chose the ISS detector based on the results obtained in Experiment 7 where ISS turned out to be the next best detector after LEFT, achieving the next highest repeatability rates on most dataset.

In order to ensure a fair comparison, we set the number of keypoints selected by the LEFT detector to 6% of the cloud of points for both Retrieval and Random View datasets, while we used 3% for the Stereo dataset. For ISS, we selected the parameters that led to approximately the same number of keypoints. As far as the parameter selection for the descriptors we used the same parameters used in Experiment IV-D.

This experiment not only allowed us to evaluate the matching capability of the proposed LGS descriptor but it also allows for a better appreciation of the role of repeatable and distinctive keypoints. In fact, using LEFT keypoint detector consistently improved the matching results in all datasets independently of the the descriptor used where it led to an increase in the recall rate ranging from approximately 5 to 12%. We attribute this to both the highest repeatability rates of LEFT but also to the fact that LEFT only select outstanding regions on objects as illustrated in Figure 2. This makes it easier for any descriptor to better discriminate different keypoints.

In addition to that, LGS still outperformed all other tested descriptors in 5 out of the 7 datasets used both when using ISS and LEFT detectors, further proving the discriminant power of LGS. It is also worth noting, that in most cases the combination of the proposed LEFT detector and LGS descriptor led to the best results overall, although in the case of the Stereo dataset the LGS still suffers from the results of lower classification accuracy as discussed in Section IV-D.

B. LEFT-LGS For Object Recognition

In this test we evaluate the effect of the keypoint detectors and feature descriptors on the actual object recognition results. To this end, we use the LEFT detector and LGS descriptor inside the object recognition pipeline while comparing to other state of the art detectors and descriptors used throughout our experimental validation.

For this test we complement the keypoint detection, description and matching steps with the additional correspondence grouping stage that aims at clustering good correspondences. In our case, we rely on the correspondence grouping algorithm based on geometrical consistency (GCG) introduced in [33].

The steps for establishing object presence or absence in a scene are summarized in the following:

1) Given a scene and a set models, first detect keypoints in both and extract the corresponding descriptor for each keypoints.
2) Establish correspondences between descriptors in the scene and in the model based on the criteria discussed in Section – i.e. The ratio between the first and second nearest neighbor. In this case, we set the ratio threshold to establish correspondences to be 0.8 based on the results obtained in Experiments IV-D and V-A. In fact, this threshold seemed to represent the best trade off between recall and precision on virtually all datasets used.
3) Once potential matches are established, apply the GCG algorithm that filters our bad correspondences and clusters good correspondences into sets that hold consensus on the presence of the same object based on geometrical consistency. The GCG algorithm relies on two main parameters:
   a) The consensus set bandwidth $\epsilon$ described in equation 9. This parameter represents the maximum distance between any two matching pairs belonging to the same object. We fine tuned the selection of...
Figure 13. Quantitative comparison of detector/descriptor pairs using the Recall vs 1-Precision metric for (a) Retrieval Dataset with noise=0.1, (b) noise=0.3, (c) noise =0.5, (d) Random Views Dataset with noise=0.1, (e) noise =0.3, (f) noise=0.5 and (g) The Stereo Dataset.

This parameter on smaller subsets of the Retrieval and Stereo datasets and decided to set $\epsilon = 100 + mr$.  

b) The number of correspondences that need to fall in the same cluster in order to declare an object to be present. Since this number trades off the number of true positives to the number of false positives we used it to produce the ROC curves presented in Figure 14. We vary the cluster size from 2% to 80% of the correspondences found in step 2.

Given the decision upon model presence or absence obtained from step 3 together with the ground truth information about this same presence/absence of the models in the used scenes, we can estimate the number of True Positives (TP), False Positives (FP), True Negatives (TN) and False Negatives (FN). We used these values to estimate the True Positive Rate (TPR), the False Positive Rate (FPR) given in equations 8 and 9 respectively. Together these values are used to plot the ROC curves reported in Figure 14. These same results are also summarized in tables I, II and III, that represent the recognition accuracy using the Area Under the Curve (AUC) metric.

$$TPR = \frac{TP}{TP + FN}$$  \hspace{1cm} (8)  

$$FPR = \frac{FP}{FP + TN}$$  \hspace{1cm} (9)
Figure 14. Object Recognition results of the different detector/descriptor pairs represented with ROC curves for (a) Retrieval Dataset with noise=0.1, (b) noise=0.3, (c) noise=0.5, (d) Random Views Dataset with noise=0.1, (e) noise=0.3, (f) noise=0.5 and (g) Stereo Dataset.

Table I

<table>
<thead>
<tr>
<th>noise</th>
<th>LEFT</th>
<th>ISS</th>
<th>LEFT</th>
<th>ISS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>97.75%</td>
<td>87.70%</td>
<td>97.99%</td>
<td>86.37%</td>
</tr>
<tr>
<td>0.3</td>
<td>91.30%</td>
<td>85.05%</td>
<td>93.35%</td>
<td>85.54%</td>
</tr>
<tr>
<td>0.5</td>
<td>88.54%</td>
<td>88.81%</td>
<td>95.90%</td>
<td>82.37%</td>
</tr>
</tbody>
</table>

Table II

<table>
<thead>
<tr>
<th>noise</th>
<th>LEFT</th>
<th>ISS</th>
<th>LEFT</th>
<th>ISS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>71.64%</td>
<td>34.05%</td>
<td>65.94%</td>
<td>33.95%</td>
</tr>
<tr>
<td>0.3</td>
<td>55.33%</td>
<td>61.76%</td>
<td>53.95%</td>
<td>66.98%</td>
</tr>
<tr>
<td>0.5</td>
<td>40.18%</td>
<td>75.74%</td>
<td>40.99%</td>
<td>66.49%</td>
</tr>
</tbody>
</table>

Table III

<table>
<thead>
<tr>
<th>Stereo Dataset</th>
<th>LEFT</th>
<th>ISS</th>
</tr>
</thead>
<tbody>
<tr>
<td>LGS</td>
<td>74.88%</td>
<td>66.75%</td>
</tr>
<tr>
<td>SHOT</td>
<td>86.70%</td>
<td>71.01%</td>
</tr>
<tr>
<td>FPPH</td>
<td>79.06%</td>
<td>72.05%</td>
</tr>
<tr>
<td>SI</td>
<td>78.57%</td>
<td>66.18%</td>
</tr>
</tbody>
</table>

These results prove the superiority of the LEFT-LGS combination that lead to the highest recognition accuracy on 6 out of 7 datasets used. In addition to that these tests highlight the importance of selecting outstanding keypoints such as the ones detected using the proposed LEFT detector. In fact, using LEFT improved the recognition accuracy of all tested descriptors. Also, as pointed out earlier in Experiment IV-D
these tests further prove that LGS and SHOT are the most robust to noise as opposed to others whose results immensely deteriorate with the addition of noise. Finally, it worth mentioning that although Random Views had overall the smallest number of good correspondences, these same correspondences were sufficient to achieve satisfactory recognition results when using the LEFT-LGS combination.

1) **Comparison with Global Descriptors:** In this experiment, the goal is to compare the performance of the proposed LEFT-LGS pair to global descriptors. To this end we used the global object recognition pipeline with two different global descriptors; namely: GFPFH and VFH. We chose VFH because it is arguably the most widely used global descriptors in the 3D literature and GFPFH as a representative of a global method relying on surface type classification. The steps followed to obtain the results of this experiment are summarized as follows:

1) Describe each model using one global descriptor.
2) Segment the scene and describe each segment. (For the segmentation we again used the Euclidean Clustering method mentioned in [30])
3) For each segment descriptor find its closest model descriptor.
4) If the closest model is not present in the scene according to the ground truth annotation then this is a false positive. Otherwise, use the ground truth transformation between the matching model and the scene and verify that the rotated model matches the segment in the scene by counting the number of points in the model that found a match in the scene.
5) If the number of points that found a match in the scene is above 70%, then this match is counted as a true positive —i.e. a correctly recognized object.

As far as the LEFT-LGS pair we used the local object recognition pipeline in order to establish correspondences between objects in the models set and the scenes. In doing so, we follow steps 1 through 3 described previously in Section V-B and we take as potential matches the models that returned the clusters with largest cardinalities. Finally, we confirm or reject potential matches following steps 4 and 5 described above for evaluating global descriptors. In other words, if the model with largest cluster cardinality is not present based on ground truth information then it is rejected. Otherwise it’s presence is further confirmed using the ground truth transformation.

In order to directly compare the performance of the proposed LEFT-LGS pair with global descriptors, we report the results of this test in terms of the percentage of correctly recognized objects across all different scenes. The results for the different datasets used are reported in tables IV, V and VI bellow.

<table>
<thead>
<tr>
<th>Retrieval Dataset</th>
<th>noise=0.1</th>
<th>noise=0.3</th>
<th>noise=0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEFT-LGS</td>
<td>81.52%</td>
<td>79.07%</td>
<td>77.07%</td>
</tr>
<tr>
<td>VFH</td>
<td>29.89%</td>
<td>28.44%</td>
<td>28.33%</td>
</tr>
<tr>
<td>GFPFH</td>
<td>87.04%</td>
<td>86.52%</td>
<td>84.22%</td>
</tr>
</tbody>
</table>

Table IV

<table>
<thead>
<tr>
<th>Random Views Dataset</th>
<th>noise=0.1</th>
<th>noise=0.3</th>
<th>noise=0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEFT-LGS</td>
<td>55.56%</td>
<td>54.44%</td>
<td>54.44%</td>
</tr>
<tr>
<td>VFH</td>
<td>15%</td>
<td>13.33%</td>
<td>13.33%</td>
</tr>
<tr>
<td>GFPFH</td>
<td>13.06%</td>
<td>13.06%</td>
<td>11.94%</td>
</tr>
</tbody>
</table>

Table V

<table>
<thead>
<tr>
<th>Stereo Dataset</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>LEFT-LGS</td>
<td>56.67%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VFH</td>
<td>8.89%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GFPFH</td>
<td>3.78%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table VI

This tests proves the advantage of relying on surface classification in describing objects where the combination of LEFT-LGS as well as GFPFH returned excellent results on the Retrieval Dataset. This confirmed the claim that using different surface types contributes in adding discrimination power to a feature descriptor. In addition, this experiment clearly demonstrate the gain brought by the hybrid nature of LEFT-LGS. This became particularly clear in datasets involving occluded objects such as the Random Views Dataset where the performance of both VFH and GFPFH saw a major drop although it involved the same objects present in the Retrieval Dataset. On the other hand LEFT-LGS was more robust although it used the global structure of the objects in detecting keypoints and describing them. We attribute this to the benefit of using hybrid ideas relying on local approaches in establishing correspondences while using global ideas in the detection/description stages that made LEFT-LGS more discriminating. It worth noting that this performance difference is even more evident on the Stereo dataset where LEFT-LGS largely surpassed both VFH and GFPFH that both failed due to the increased clutter in this dataset.

VI. **Conclusion and Future Work**

This work addressed the false dichotomy between local and global approaches present in the field of Object Recognition. First, this research proposed a novel criterion for robust 3D keypoint detection that addresses most of the difficulties inherent to the task of keypoint selection —i.e the detection of keypoints that are: (i) located on outstanding regions across the entire object, (ii) Robust to noise and (iii) Repeatable. Most importantly, we introduced a novel way of looking at
the problem globally rather than locally through our LEFT criterion. Our results clearly highlight the relevance and robustness of the proposed approach for 3D keypoint detection, especially in terms of getting away with thresholds for saliency detection. Second, we introduced a novel signature-based 3D keypoint descriptor that bridges the gap between local and global descriptors. The results highlighted the benefits of using signatures and their role in avoiding great loss of information – as opposed to what happens with histogram-based approaches.

Finally, we showed that the relative positions of the keypoints with respect to local and global support regions hold enough discriminating power while they replace low-level features such as point normals, which are very sensitive to noise. The advantages of the hybrid LEFT-LGS combination became clear when we compared them to both local and global SOTA approaches, where LEFT-LGS achieved better results on most benchmarks used in this work. However, during the course of this research we identified several directions for future work that could take the proposed algorithms one step further. As far as the proposed LEFT detector, the main focus would include addressing the problems faced with highly cluttered scenes. Ideally, the proposed global approach should become independent of the segmentation step. In the case of the LGS descriptor, the most important aspect requiring further investigation would be addressing the problems caused by data acquisition and sensor noise when it comes to misclassification of points. For example, a different method for classification based on primitives should be part of a future work in this research. In fact, we believe that a better surface classification scheme can lead to much more robust signatures as demonstrated in Figure 12 (g). Also, a method for automatic selection of the number of classes $N$ should further improve the obtained results with LGS.

REFERENCES


Isma Hadji All about you and the what your interests are. Don’t forget to put your name in between a pair of {}’s that are set as raw \TeX.

G. N. DeSouza Same again for the co-author.