Technical Report No. 2013-614

3D Object Recognition using Local Shape Descriptors [*]

Mustafa Mohamad

School of Computing
Queen's University
Kingston, Ontario, Canada

mustafa@cs.queensu.ca

November 7, 2013

[*]This technical report is based on the author's depth paper.

Abstract

3D object recognition is a challenging problem with important applications such as robotic perception. The most promising approach to solving 3D object recognition is through solving the correspondence problem. The goal of the correspondence problem in the context of 3D object recognition is to find correspondences between the objects to be recognized and the scene. If correspondences exist between an object and the scene it can be hypothesized that this object exists in the scene. Once these hypotheses are verified objects are recognized. The most common approach to solving the correspondence problem for 3D object recognition has been through techniques that try to find correspondences between local regions of the models and the scene. In this report, we focus on different local techniques and highlight their weaknesses and strengths as well as provide directions for future research.

Table of Contents

List of Figures

List of Tables

1 Introduction

The holy grail of computer vision research is to be able to build a system that can perform all the vision tasks that a human being can perform at the same level of ease and efficiency. One of those vision tasks that humans can perform with ease but machines find great difficulty in performing is accurate and efficient object recognition. Object recognition is considered to be one of the high level vision problems and has enjoyed active research for the past 4 decades [4]. Generally, the object recognition research community has been split into two camps: Those who deal with 2D images and those who deal with 3D pointclouds or meshes. 2D images are created by projecting the scene onto a plane by capturing the light intensity detected at each pixel. Alternatively, 3D pointclouds capture the 3D coordinates of points in the scene. The main difference between the two types of data is that 3D data includes depth information whereas 2D does not. Historically, the community that studies 2D object recognition has been larger. However, this is changing due to the fact that new technology has enabled acquisition of 3D data using cheaper sensors such as RGB-D cameras. RGB-D cameras such as the Microsoft Kinect capture a regular colour image (RGB) along with the depth(D) associated with each pixel in the image.

3D object recognition techniques can be roughly categorized into global and local techniques. The focus of this depth report is study and compare the local techniques because they are more general and have received more attention in recent literature. The report is organized as follows: Section 2 defines the 3D object recognition problem, briefly discusses global techniques and then introduces the correspondence problem and its relationship to 3D object recognition. Furthermore, it discusses some historical techniques for solving the correspondence problem for 3D object recognition. Section 3 which is the main section of the report surveys some well known and recent local techniques for solving the correspondence problem in 3D object recognition and provides a comparison of these techniques. Finally, Section 4 looks at future research directions for improving the state of the art in the area.

2 Background

2.1 Defining 3D Object Recognition

Given a database of 3D models (objects), object recognition is the problem of finding all instances of the database models in an arbitrary scene as well as determining the pose of the detected objects. The pose of the object is the rigid transformation that aligns the object in the database to the object's instance in the scene. In the rest of the report, rigid transformation and pose will be used interchangeably. In 2D, the rigid transformation has three components and they are two translations in each of the x, and y directions and a rotation in the xy-plane. In 3D, the pose has six components which are the translations in each of the of x,y, and z directions as well as a rotation about each of these axes, namely, the pitch, yaw and roll. Therefore solving the problem of 3D Object recognition is the problem of finding a known object in the scene along with its 6D pose.

3D data can be captured using a variety of sensors including stereo cameras, time of flight laser scanners such as LiDARs, as well as infrared sensors such as the Microsoft Kinect or Panasonic DI-Imager. All sensors can only capture a single view of the object with a single scan. This view is referred to as a 2.5D scan of the object. Therefore to capture the entire 3D shape of the object the

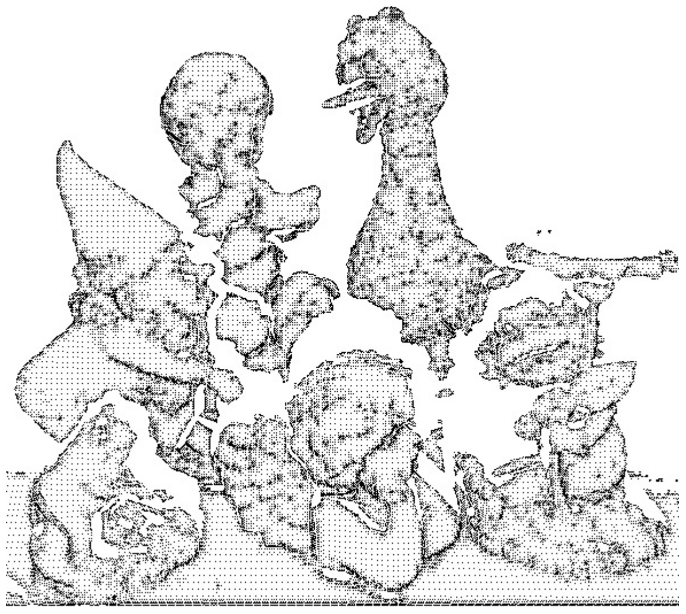

Figure 1: A typical 2.5D scene showing both clutter and occlusion captured using a vivid LIDAR scanner. Picture courtesy of [1]

sensor captures multiple instances of the object at different viewpoints. In 3D object recognition the goal is to recognize a full 3D object stored in the model database in a 2.5D scene where only part of the object is visible. In practice, there are other issues that add to the challenge of solving the problem of 3D object recognition and they include:

- *Occlusion*: A part of the 3D object is always hidden due to self-occlusion or occlusion by other objects in the scene. Formally, occlusion is defined as:

$$\text{occlusion} = 1 - \frac{\text{model surface patch area in the scene}}{\text{total model surface area}} \tag{1}$$

- *Clutter*: A scene may include many closely spaced objects, making it difficult to determine the source object of a data point. Formally, clutter is defined as:

$$\text{clutter} = 1 - \frac{\text{model surface patch area in the scene}}{\text{total scene surface area}} \tag{2}$$

- *Noise*: Sensors are not perfect and therefore a 3D representation of the same view of an object is never exactly the same and can possibly include missing parts depending on the quality of the sensor.

- *Sampling Resolution*: The 3D data of objects in the database might be captured using a different sensor with a different sampling rate than the one used to capture the 3D scene data.

Figure 1 shows a typical 2.5D scene with clutter and occlusion.

2.2 Global Shape Descriptors

There are several techniques in the literature that describe an entire object using a single global description. Although these methods tend to be very efficient, they are very sensitive to occlusion and clutter because objects in the scene are not isolated and therefore cannot be described globally. These methods tend to be used in scenes that contain no clutter and only self-occlusion or for solving the problem of *Object Class Recognition* whose goal is to identify objects of the same class as an object in the database. Object Class Recognition generally deals with scenes that contain only a single object that is not occluded or cluttered with other objects and therefore global shape descriptors tend to do well here. Osada *et al.* [5] proposes *shape distribution* which is simple global shape descriptor that randomly samples pairs of points on the object and calculates the distance between each pair. The distances are placed into a histogram based on the distance interval that they fall into. The histogram of point pair distances represents the shape distribution of the object and is used to compare to a scene object's shape distribution using the L_1 norm. Their method was only successful in identifying simple objects two thirds of the time. Wahl *et al.* [6] presents a similar shape distribution method but instead of calculating only the distance between a pair of points, the authors use a pair of *oriented* points and calculate three other properties of the pair in addition to the distance property by exploiting the normal information. An oriented point is a point whose normal is known. Their shape distribution is therefore a 4D histogram. Their technique achieved 80% recognition for single object scenes with up to 45% occlusion. The technique was also shown to be robust to different sampling resolutions but sensitive to noise. Hetzel *et al.* [7] proposes another global histogram technique that builds a multi-dimensional histogram of the object based on depth, surface normals, and curvature properties of the object. These properties are calculated at local regions on the object and values are binned into a single multi-dimensional histogram. The method achieved a recognition rate of 91% on 30 database objects with up to 40% occlusion.

The Potential Well Space Embedding proposed by Shang and Greenspan [8] is another global technique that takes a different approach to describing an object. The *error surface embedding* global descriptor takes advantage of some of the properties of the Iterative Closest Point (ICP) algorithm proposed by Besl and McKay [9]. The goal of the ICP algorithm is to find the best alignment between two sets of points. The algorithm attempts to achieve this by iteratively minimizes the collective distance between pairs of points across two point sets until no improvement can be made. It requires an initial alignment which is close to the optimal solution in order for it to converge to the global minimum; otherwise it will converge to a false local minimum. Their technique takes advantage of the different local minima that the ICP algorithm can converge to. Their error surface embedding descriptor is constructed using the following steps:

1. Generate a random generic model, $\mathcal{M}$, that is at least as complex as the most complex object and at least as large as the largest object in the database. This is achieved by building an object composed of 120 spheres of random radii and location within a bounding box at least as large as the largest object in the database.

2. For each model view, the error surface model is calculated by first running ICP to register it to the generic model, $\mathcal{M}$. The rotation and translation relative to the original view position is used as the origin of the error surface.

3. The view is perturbed to a set of 30 translational positions and ICP is applied to register each of the perturbed views to $\mathcal{M}$. The rotation and translation of each registration relative to the

origin is stored. The 30 pairs of rotation and translation values of the ICP registration form the error surface embedding which describes the object globally.

The technique was found to be very robust to noise and data sparseness. It achieved a recognition rate of 97% on a database of 60 objects without reporting occlusion rate. They also achieved an impressive 87.8% on a standard Object Classification database where objects of the same class were recognized. However, as a global method, the technique was ineffective in dealing with high occlusion or clutter.

2.3 The Correspondence Problem

As outlined above using Global Shape Descriptors to solve 3D Object Recognition is not well suited for real world situations where high occlusion and clutter can exist. Another approach to solving object recognition comes through solving another well known problem in computer vision, namely, the *correspondence problem*. Given two different point sets that share a common subset of points, the correspondence problem is to find matches between points in the first point set and points in the second point set. An example of a correspondence problem is image-stitching where two overlapping views of the same scene are stitched together to create a wide-angle view of the scene. Another is 3D model construction where overlapping views of a 3D object are obtained and registered in the correct order in order to build a full 360° view of the entire model. Simultaneous Localization and Mapping (SLAM) is a technique used by robots driving in unknown environments to build a map of the environment as well as keep track of their location relative to their starting position. SLAM uses the correspondence between overlapping frames (images) of the environment to compute the transformations that describe the robot's motion, as well as to register the frames to build the map. Finally 3D Object Recognition can also be posed as a correspondence problem where the goal is to find the points in the scene that correspond to the points of different models in the model database.

In order to determine the 6D pose of a model in the scene, it is sufficient to find 3 model to scene correspondences. The two sets of triplets are used to compute two reference frames, and the rigid transformation between the two frames is computed and used to transform the model to the scene. A RANdom SAmple Consensus (RANSAC) algorithm [10] is generally used for finding the best corresponding triplets that describe the transformation. Algorithm 1 describes the RANSAC technique:

The complexity of RANSAC for determining the 6D pose of an object in the scene is $\mathrm{O}\big(|S|^3\big)$ where $|S|$ is the number of scene points. It is clear that this algorithm is inefficient and therefore recent object recognition techniques do not attempt to establish correspondences using raw data points but use more complex primitives that reduce the correspondence search space prior to applying RANSAC. Therefore, RANSAC is used in the Object Recognition Pipeline as away to probabilistically find the best correspondence within a set of potential correspondences. A more efficient RANSAC algorithm, termed the 4 Point Congruent Point Sets (4PCS) algorithm, proposed by Aiger *et al*. [11] has a complexity of only $\mathrm{O}\big(|S|^2\big)$. The algorithm uses 4 co-planar points as the basis to construct the local reference frame instead of a 3 point basis as with vanilla RANSAC. The 4 co-planar points, $\{a, b, c, d\}$ are chosen such that two intersecting line segments, ab and cd, are formed. Under rigid transformation, the following two ratios are preserved:

$$r_1 = \frac{||a-e||}{||a-b||} \qquad r_2 = \frac{||c-e||}{||c-d||} \tag{3}$$

Algorithm 1 Random Sample and Consensus Algorithm for 3D Pose Determination

```
$K \leftarrow$ max number of iterations
$v \leftarrow$ minimum number of model to scene correspondences threshold
$P_M \leftarrow$ 3 ordered points randomly selected from $M$
$T_{best} \leftarrow$ empty
$e_{best} \leftarrow \infty$
$k \leftarrow 0$
while $k < K$ do
    $P_S \leftarrow$ 3 ordered points randomly selected from $S$.
    Compute the transformation, $T_k$, between $P_S$ and $P_M$
    $v_k \leftarrow$ number of model to scene correspondences after $T_k$ is applied to $M$
    if $v_k > v$ then
        $e_k \leftarrow$ transformation error of $T_k$
        if $e_k < e_{best}$ then
            $T_{best} \leftarrow T_k$
        end if
    end if
end while
```

This property is used to eliminate many false matches when searching for a matching 4 point basis in the scene and thus resulting in a more efficient algorithm. A technique that serves the same purpose as RANSAC is the General Hough Transform (GHT) proposed by Ballard [12]. Although (GHT) is not probabilistic and therefore is guaranteed to find the correct solution its complexity is worse than RANSAC as well as requiring an exponential storage space in the number of parameters estimated. The parameters are used to describe the pose of the object. For 3D Object Recognition, seven parameters need to be estimated [13]. Due to this reason, RANSAC is used more often in practice. Some of the historical approaches in solving the correspondence problem in the realm of 3D Object Recognition will be highlighted prior to delving into more recent techniques in solving the problem.

2.4 Historical Approaches

One of the earliest techniques for 3D object recognition is the *Interpretation Tree* proposed by Grimson and Lozano-Perez [14], [15]. In their work, the models used are composed of a set of surface patches called faces. Each node in the tree stores a correspondence between a model face and a scene point. At the root of the tree no correspondences are stored. At each new level of the tree a new model face is chosen and its correspondences with all scene points are stored. Once the tree is built, the longest paths in the tree represents the most likely hypothesis for the model's 6D pose. This results in a tree with $|M|^{|S|}$ leaves for a model M and the scene S. This prohibitive exponential size of the tree is pruned through the use of rigidity constraints that reduce the number of nodes that are added at each level. An example of such a rigidity constraint, is the distance constraint, which states that if two model faces correspond to two scene points the distance between the model faces must be very similar to the distance between the scene points. Other constraints based on normal orientation are also utilized for further pruning. To handle occlusion, a null node is added at each level in the tree. The null node represents the possibility that a model face does not correspond to any scene point due to part of the object being invisible due to occlusion. Although pruning might help reduce the size of the tree, the technique is still an exhaustive search which is exponential in complexity. These early works dealt with very small datasets and scenes composed of tens of points and thus the technique was viable for these simple datasets. The Interpretation Tree, however, can be applied to match more complex primitives such as *Local Shape Descriptors*; this will be elaborated upon in the next section.

Geometric Hashing is a general technique applicable to both 2D and 3D object recognition. For the 2D case, in the offline phase, a representation of a model, M_i, is built by picking two points, (m_1, m_2), as a basis for M_i. A reference frame is constructed using the basis by setting the midpoint of the (m_1, m_2) as the frame origin and the vector $\overrightarrow{m_1 m_2}$ as the x-axis. The x-axis is rotated 90 degrees clockwise to get the y-axis. In the next step, the coordinates of all other model points relative to this reference frame are computed and used as a key into a 2D hash table where the $(model, basis)$ pair, $(M_i, (m_1, m_2))$, is entered. This operation is repeated for all possible ordered basis pairs of M_i. The same is done for other models in the database. In the end of this phase, the hash table contains entries for all models and for all possible bases for each model. In the online recognition phase, two points are randomly picked from the scene and used as the basis to express all other scene points. The coordinate of a point in the basis reference frame is used to access the appropriate hash table cell, and a vote is cast for all entries in the cell. The operation is repeated for all possible scene ordered basis pairs. For $(model, basis)$ pairs that get a significant number of votes relative to other $(model, basis)$ pairs, the transformation of the model to the scene that results in the best least-squares match between all corresponding points is computed. This transformation is used to transform the model to the scene to verify that an acceptable portion of the model correspond to the scene. The verification procedure in object recognition is further discussed in the upcoming sections. Geometric Hashing can be extended to 3D object recognition with 6D pose determination by using a basis composed of 3 non-collinear points. The complexity of the online phase of geometric hashing is $\mathrm{O}\big(|H||S|^{k+1}\big)$, where $|H|$ is the bin occupancy of the hash table, $|S|$ is the number of scene points, and k is number of points that form the basis. For 3D, this would be $\mathrm{O}\big(|H||S|^{4}\big)$. During the offline phase, re-hashing of the hash table can improve access times up to $\mathrm{O}\big(1\big)$. Another interesting aspect of geometric hashing is that hash table bins that contain a large number of entries can simply be ignored in the online phase because their information is not discriminating enough. An extension of this idea leads to weighted voting where the bin's information (weight) is inversely proportional to the size of the bin.

Greenspan [16] proposes the *Sample Tree*, a hypothesis-test-verify approach. Given a set of models in the database, each with its own set of discrete model poses that approximate the infinite pose space of a 3D object, a binary tree classifier, the Sample Tree, is built for each model in the offline phase such that poses are eliminated as the depth of the tree increases. To build the sample tree offline, an initial hypothesis is formulated that a point $s_0 \in S$ corresponds to a point in M, and all poses that satisfy this hypothesis are stored at the root. At this stage, the hypothesis does not establish a correspondence between s_0 and a particular point in M. Since a scene point can correspond to any point on the model, all poses are stored at root. Next another point, s_1, in the scene is hypothesized to correspond to a point on the model. All poses that intersect with both s_0 and s_1 are added to the right child of the current node, the "true" child. All poses that do not intersect with both s_0 and s_1 are added to the other child, the "false" child. Although s_0 has not been fixed on the model, the relative position of s_1 to s_0 forms a vector that might not intersect with some poses and this is how poses are filtered out. The tree is continually built in this binary fashion by adding more hypotheses until a small enough number of poses remain in each of the leaves. In the online phase, all scene points are processed through the sample tree of each model returning various potential poses. The poses are verified and only verified poses are accepted.

Both of Geometric Hashing and the Sample Tree can be applied after an *interest point* extraction phase. An interest point is a point on the surface of a pointcloud or mesh that exhibits geometric variety exists in its local neighbourhood. Another term used for interest points is *salient* points and these two terms will be used interchangeably throughout the report. An example of an interest point is a local maxima, minima or an inflection point of the 3D surface. A performance evaluation of 3D interest point extraction techniques is presented in [17]. Clearly interest point extraction would speed up both of the aforementioned algorithms due to a reduced point set.

3 Solving the Correspondence Problem Using Local Techniques

Object recognition using Local Shape Descriptors(LSDs) has shown the most promise in handling occlusion and clutter in the scene [18], and therefore has received much attention. We outline the basic framework that is used in techniques that rely on LSDs for 3D object recognition. In the offline phase, a model database is constructed by computing LSDs for each object to be recognized. The LSD encodes certain geometric properties of local parts of the object. At the end of the offline phase, each object in the model database is represented by a set of LSDs. In the online phase, LSDs are calculated for points chosen from the scene and are matched against LSDs stored in the database. The matches generate a set of k hypotheses for both the candidate object and the candidate pose, $H = \{h_1, h_2, \ldots, h_k\}$. Each $h_i = (M_i, T_i)$ where M_i is the candidate object model and T_i is the candidate rigid transformation that represents the pose. These hypotheses are verified during a hypothesis verification stage and invalid hypotheses are rejected. For every accepted hypothesis, the object model M_i is transformed to the scene using T_i and then a refinement step is applied which further improves the alignment of M_i with the object in the scene. This refinement step is accomplished using the Iterative Closest Point (ICP) algorithm proposed by Besl and McKay [9] or a variant of it. Since T gives a very good initial alignment of the object to its instance in the scene ICP tends to converge to the global minimum which represents the best alignment between the two point sets. Figure 2 summarizes the pipeline used for object recognition using LSDs. As with the historical techniques the LSD representation can be calculated for specific interest points only, or by uniformly sampling the point sets, or by using all points. Usually, such choices depend on the resolution of the data being processed.

3.1 Types of Local Shape Descriptors

In this section we describe the LSD formulations found in the literature as well as how LSDs are matched. Prior to discussing the literature, we defined some preliminary notation. An oriented point, p, is a point with an associated normal, $\mathbf{n}_p$. Let the local neighbourhood of p, $N_r(p)$, be defined as the set of points which fall within a sphere of radius r, centred at p. We will sometimes use the terms *support*, *support region*, *support distance* to refer to p, $N_r(p)$, and r, respectively. The condition that all points must be within a distance r from p to be in $N_r(p)$ is referred to as *support*

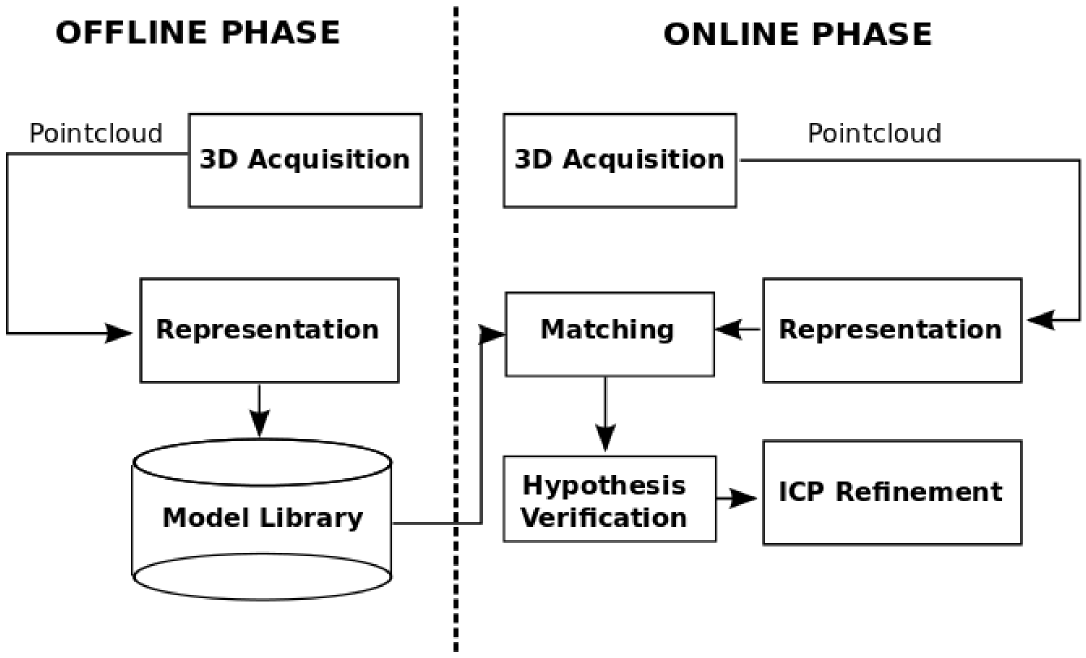

Figure 2: 3D Recognition Pipleline

distance constraint. For a point q with normal $\mathbf{n}_q$, the *support angle* is the angle between $\mathbf{n}_p$ and $\mathbf{n}_q$. The sets of points in $N_r(p)$ must also satisfy the *support angle constraint* which places an upper bound on the angle between $\mathbf{n}_p$ and $\mathbf{n}_q$. The reason for the above constraints will be discussed later on. A note about notation, we will use the notation m_i to refer to a support point in a model, and s_i to refer to a support in the scene; otherwise, we will simply use p to refer to a generic support point that could belong to both the model and the scene.

3.1.1 Local Neighbourhood Descriptors

The most well known local shape descriptor is arguably the *Spin Image* descriptor proposed by Johnson and Hebert [2] The authors compute the Spin Image LSD for every oriented point on the surface of an object and the scene. The Spin Image LSD is a 2D histogram which measures two distances of every point in the neighbourhood of p, $N_r(p)$ relative to p. Given a point $q \in N_r(p)$, the first distance calculated, α, is the perpendicular distance of q to the line going through $\mathbf{n}_p$. The second distance, β, is that of q to the tangent plane of p. See Figure 3. These distances are placed into a 2D histogram quantized by the parameters α and β. LSD comparison between two Spin Images is performed by calculating the linear correlation between them. The authors also discuss the issue of choosing the best support distance and support angle for the Spin Image descriptor. The larger the values of the support distance and support angle the more descriptive the Spin Image but the less robust it is to occlusion and clutter. The is due to the fact that when calculating the LSD in the scene using a large support distance and angle there is a good chance that the support region will be corrupted because of missing points due to occlusion or include points belonging to another object due to clutter; this in turn corrupts the LSD computation. The authors determine these two values experimentally.

To generate correspondences between the scene and the model database, scene points that

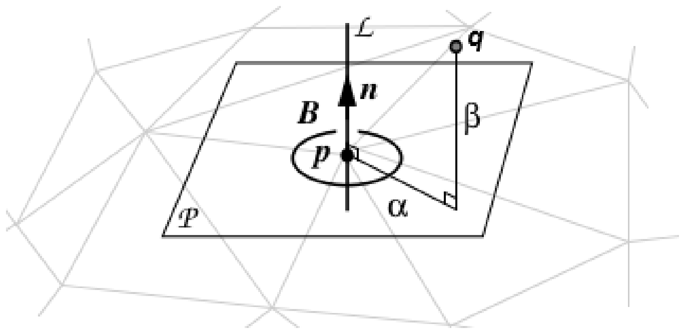

Figure 3: Distances calculated for the Spin Image Descriptor. Picture courtesy of Johnson and Hebert [2]

adequately cover the scene surface are sampled. The number of selected scene points can range from one tenth to one half of the scene points depending on the complexity and amount of clutter in the scene. Furthermore, since the Spin Image LSD does not encode a local reference frame, a single correspondence is not enough to find the pose of the model in the scene. Three correspondences are required in order to compute a local reference frame of the model in the database and in the scene. The pose of the model in the scene is simply the rigid transformation that aligns the two local reference frames. Every scene point's LSD is compared to all LSDs in the model database and a histogram of the similarity score between the scene LSD and model LSDs is created. The upper outliers of this histogram are chosen as valid correspondences. The upper outliers of the similarity histogram are detected by considering all correspondences whose similarity measure is greater than the upper fourth plus three times the fourth spread. The authors mention that the matching approach of only considering upper correspondence outliers is reliable for two reasons: The first is that if no outliers exist then this scene point is not descriptive enough to distinguish a specific object. The second is that the existence of multiple outliers implies multiple model/scene point correspondences which should all be checked in the verification stage.

The set of correspondences are further filtered by exploiting geometric consistency between the correspondences. Let $C_i = \{m_i, s_i\}$ be the i^{th} correspondence where m_i and s_i are the corresponding model and scene points respectively. If two correspondences, C_1 and C_2 are part of the same model in the scene and in the model database then it is expected that the correspondences will be geometrically consistent. In other words, the difference in position and normal orientation of the points in the scene, (s_1, s_2), is very similar to the difference in position and normal orientation of the model points, (m_1, m_2). Correspondences that are not geometrically consistent with at least a quarter of the entire set of correspondences are filtered out. Finally, correspondences grouping is performed where correspondences that are geometrically consistent and far apart are grouped together. Choosing correspondences that are far apart is important since correspondences that are close together generate transformations that are susceptible to noise in point position [19]. Given the list of correspondences, L, with n correspondences, a correspondence, C_i, is used to initialize a group $G_i = \{C_i\}$. Next, a correspondence $C_j \in L$, is added to G_i if it is geometrically consistent and far apart with respect to all correspondences in G_i. This is done for $j = 1 \dots n$ where $j \neq i$. Correspondences are allowed to belong to more than one group to handle model symmetry. Once correspondences are grouped, the best rigid transformation, T, that aligns the corresponding point sets is calculated [20]. The transformation minimizes the following error function:

$$E_T = \sum ||s_i - T(m_i)||^2 \tag{4}$$

Correspondence grouping serves two purposes: First it avoids the combinatorial explosion resulting from randomly selecting three correspondences to generate a transformation which is generally what is done using a vanilla RANSAC approach to find the best correspondences. Second, it generates more reliable transformations because they are based on a larger number of correspondences. The transformations produced are verified by applying them to the associated models and checking the proportion of model points that are explained by the scene points after the transformation. In other words, the proportion of inliers, i.e., model points that have a corresponding scene point whose distance to the model point is less than a specified threshold. This threshold tends to be set to two times the resolution of the mesh or pointcloud. If the proportion is higher than a specified threshold the hypothesis is accepted. The authors find that their technique performs well up to 70% occlusion and find that occlusion affected performance more than clutter.

The authors extend their work in [21] by presenting a compressed form of the Spin Image representation in order to speed up the matching process. The library of all model Spin Images is compressed using Principle Component Analysis [22] to find the most representative Spin Images. Next, all other Spin Images are projected into the eigenspace spanned by the most representative Spin Images. During matching, a scene Spin Image is also projected into the eigenspace and matched to its closest neighbour in the eigenspace in terms of l_2 distance. PCA Spin Images was shown to be much more efficient than regular Spin Images with an acceptable reduction of recognition performance.

Ruiz-Correa *et al.* [23] proposed the spherical Spin Images representation in which Spin Images that were highly similar were first mapped into the same equivalence class. The equivalence classes are then mapped into unit vectors on the unit sphere. The spherical representation maps the linear correlation between two Spin Images to the cosine of the angle of the two unit vectors that represent the Spin Images. They also propose a method to compress their spherical Spin Images representation by using a random projection technique [24] instead of PCA. The PCA compression is affected by the Mesh resolution while the random projection compression is independent of the mesh resolution [25]. In addition the complexity of random projection is $O(d^2n)$ while PCA has a complexity of $O(n^3)$ where n is the number of Spin Images and d is the number of dimensions of the basis of the compressed space which tends to be much less than n. The authors showed that the random projection representation is more accurate and efficient than the PCA representation.

The Spin Image descriptor essentially projects all points that fall onto a cylinder of specific height and thickness onto a single point in a 2D histogram. Therefore there is dimensionality reduction in the descriptor that naturally leads to a loss of information in the description of the local neighbourhood. The *3D Shape Contexts* LSD proposed by Frome *et al.* [26] builds a local shape descriptor by partitioning the spherical support region around a 3D point into bins. For each point that falls within a bin, a weighted contribution of the point to the score of the bin is computed. Essentially, the authors compute a 3D histogram as opposed to a Spin Image which is 2D histogram and show that their LSD is more descriptive. The spherical support is partitioned by dividing the elevation and azimuth dimensions into equally spaced boundaries and the radial dimension into logarithmically spaced boundaries. The logarithmically spaced boundaries of the radial direction create smaller bins near the support and larger bins as one moves further away from the support. The reason for the logarithmic division in the radial direction is to make the descriptor more robust to shape distortions that occur further away from the support. The spherical support is placed at a 3D point so that the north pole of the sphere aligns with the normal of the point. This alignment leaves a degree of freedom in the azimuth dimension, which must be dealt with in order to ensure that the LSDs computed for scene points match LSD computed on the model. This is taken care of by computing D different LSDs for each model point where D is the number of division in the azimuth direction.

In the offline and online phase, points are chosen randomly to compute the model and scene LSD representations. In the matching phase, a *representative descriptor cost* is calculated for matching a model, M_i, to the scene, S. Given that M_i is represented using a LSD set calculated at points $\{m_1, \ldots, m_L\}$ and the scene is represented by an LSD set calculated at points $\{s_1, \ldots, s_K\}$: The representative descriptor cost, $cost(M_i, S)$, is calculated using the following expression:

$$cost(M_i, S) = \sum_{k=1}^{K} \min_{l=1}^{L} dist(m_l, s_k) \tag{5}$$

where $dist(m_l, s_k)$ is the similarity between the LSD of m_l and that of s_k. The representative descriptor cost measures the similarity between the scene and the model, M_i. Models that have significant similarity to the scene are recognized. Using their dataset, their descriptor achieved a recognition rate of 49% while Spin Images achieved only 34%. The reason for these low recognition rates is that the dataset was composed of different cars models which were highly similar in shape.

Another issue with the Spin Images descriptor is that it does not encode a local reference frame. This implies that one correct correspondence between the scene and a model is not enough to determine the pose of the model in the scene. At least three correspondences are required to build a local reference frame for both the scene and the model. Given two local reference frames, the transformation between them can be computed to find the pose of the model in the scene. In 3D Shape Contexts, a single correspondence is enough to determine a candidate pose of the object. However, their LSD uses an ambiguous local reference frame which required the calculation of D different version of the descriptor. On their dataset, the number of model LSDs was increased from 83,640 to 1,003,680 in order to handle the degree of freedom in their reference frame, where D was set to 12. Mian *et al.* [3] propose a descriptor that reduces the ambiguity of the reference frame to only 2 versions. Their proposed LSD is for meshed 3D objects, called a *Tensor*, is constructed using the following steps:

1. Decimate the mesh to allow for faster processing.
2. Randomly select two vertices that satisfy distance and angle constraints determined experimentally.
3. Define a 3D coordinate frame on the two vertices (v_1, v_2) chosen as follows:
   (a) Origin of frame is the midpoint of the segment (v_1, v_2).
   (b) z-axis = average of the normals of v_1 and v_2.
   (c) x-axis = cross product of the two normals is the x-axis.
   (d) y-axis = cross product of x-axis and z-axis.

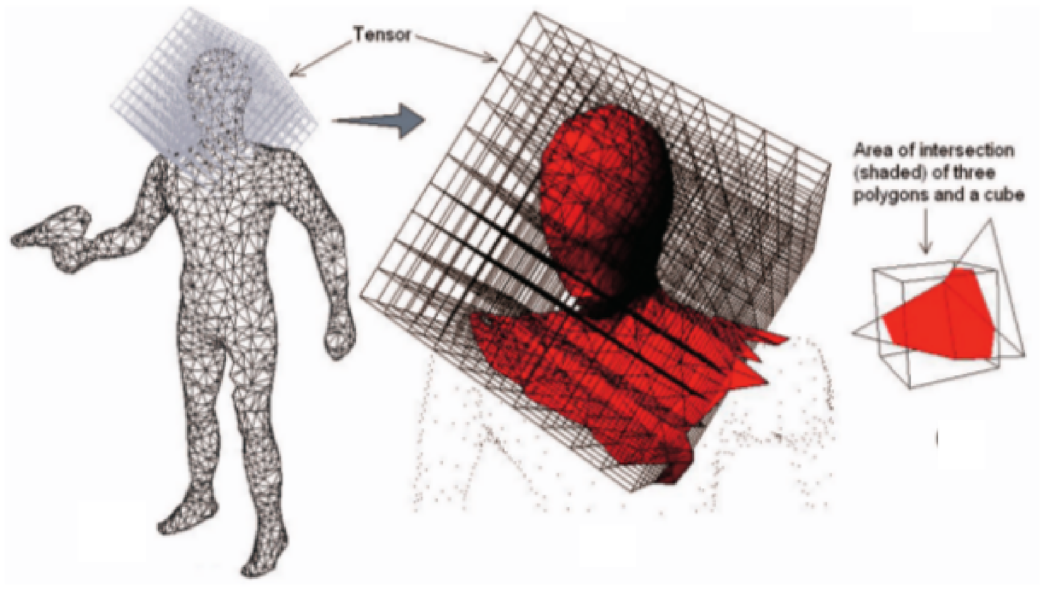

Figure 4: Constructing a Tensor by placing a grid at the origin of a local reference frame and computing the area of intersection between the bins of the grid and the mesh. Picture courtesy of Mian *et al.* [3].

4. Construct a local 3D grid whose origin is the bases frame of the previous step
5. In each bin of the grid, calculate the surface area of intersection of the bin with the mesh using the polygon clipping algorithm [27]. This 3D grid is called the Tensor.

Figure 4 shows the steps of constructing the Tensor. The Tensor is a 3D histogram which also stores a local reference frame which is invariant to rigid transformations. However, there is an ambiguity in the Tensor description of the same pair of points. Using the ordered pair (v_1, v_2) to compute the local reference frame results in a different local reference frame than using the ordered pair (v_2, v_1). This ambiguity is handled of by storing both versions of the Tensor.

In order to make the matching process more efficient a 4D hash table is used to store the Tensors of the model database. Given a computed Tensor, each non-empty bin contributes an entry (a, b) to the appropriate hash table cell where a and b are the indices of the object and Tensor, respectively. The four dimensional index of the appropriate hash table cell is composed of the i, j, k coordinates of the bin and the angular difference, θ_d, between the pair of normals of the pair (v_1, v_2) used to construct the Tensor. The authors report a recognition rate of 96.5% for their algorithm versus 87.8% for Spin Images for objects with occlusion of up to 84%. The database used in this work is regarded as the most popular object recognition benchmark for 3D object recognition [28] and it will be referred to as the *Standard 3D Object Recognition Benchmark* from this point onwards. The database contains 50 scenes of 5 free-form models with various levels of occlusion and clutter. The maximum occlusion rate for this dataset is 91.4%.

Scale space analysis is a technique well-established in the realm of 2D interest point detection. In 2D, it is used to detect interest points that are invariant to scale. In other words, an image of the same object taken from a close distance and a far distance should have the same interest points. In 2D, the distance of the camera to the object effects the size (scale) of the object in the image and therefore affects interest point detection. Given a 2D object image, scale space analysis simulates different scales by down sampling and smoothing the image at several levels. The idea is that downsampling and smoothing is analogous to a camera moving away from the object where the number of pixels (level of detail) that represents the object decreases. Interest points that can be consistently detected across different levels of detail are chosen as the scale-invariant interest points. However, scale is not an issue in 3D, because 3D sensors give the same metric measurements of the point position in space no matter the distance to the sensor. Distance to sensor in 3D could mean a sparser or noisier point cloud but still at the same scale. Although scale is not a problem in 3D interest point detection, Novatnack and Nishino [29] propose a technique to find scale-dependent interest points. This is the opposite goal of scale-space analysis for the 2D case. In their follow up work [30] which is further developed by Bariya *et al.* [31] the authors build scale-invariant LSDs for each of the scale-dependent interest points and use the scale as a filter in the matching stage of a 3D object recognition pipeline. Essentially, an LSD computed from the scene only queries database LSDs that are based on interest points detected at the same scale. This idea improved both efficiency and recognition accuracy. Furthermore, scale-space analysis also gives information about the inherent or natural support distance of the interest point and therefore unlike previous works the support distance is automatically determined. In order to perform scale-space analysis for a 3D mesh, the authors first parameterize the 3D mesh using a 2D plane and then map each of the 2D points of that plane to the normal of the corresponding 3D point to create a normal map. The normal map is then used to perform scale-space analysis as in the 2D case to detect scale-dependent interest points. To build the descriptors, the authors first compute a rotation-invariant reference frame based on the geodesic polar coordinates of the points in the support region and re-express each of the points in the support region in terms of this reference frame. Once the points are re-expressed, the normal of each point is stored at its new location resulting in a normal field; this descriptor will be referred to as the *Normal Field*. In order to compare their LSDs the authors use the cross-correlation between the normal fields as a similarity measure.

In the online recognition phase, the authors use an Interpretation Tree [14], to match the model LSDs to their scene LSDs. As discussed previously, the Interpretation Tree is an exponential data structure, however, the authors use the LSDs which are based on interest points as the primitives to match, instead of the raw points which greatly reduces the search space. In addition, the authors employ constraints to bound the size of the Interpretation Tree to achieve efficient matching. As previously detailed, at each new level of the tree the nodes store the correspondences between a model LSD and all scene LSD that match the model LSD. The first constraint used in establishing these correspondences is the scale associated with the LSD where LSDs are only matched with LSDs of the same scale. Furthermore, the tree is built in a hierarchical fashion where going deeper in the three corresponds to finer scales in the scale-space. This is inspired by the intuition that interest points detected at coarser scales, represent geometric variations that are more prominent in size, and therefore are associated with LSDs that hold relatively greater discriminative information. Another constraint employed, is that each level of the tree stores only the top 5 matches to the scene. In addition, a geometric constraint is also applied where a correspondence is added to the tree only if it is consistent with the rigid transformation of its potential parent node.

Once the interpretation tree is built, the top 20 hypotheses in the last level of the tree are verified using the area of overlap between the scene and the model. The best hypothesis out of the

20 is refined using ICP and the scene points matching the transformed model are removed from the scene. A new interpretation tree is built for the next model and the process continues until 2 or less scale dependent interest points remain in the scene. The authors achieve a recognition rate of 97.5% on the Standard 3D Object Recognition Benchmark which slightly outperformed Tensors [3].

Zhong [32] proposes the Intrinsic Shape Signature (ISS) LSD. The authors perform PCA analysis a model's points and use the ratio between principle components to detect salient point. Points that exhibit a significant difference between their principle components are considered salient because they exhibit changes in their support region as opposed to having a relatively flat support region. The principle components for each of the salient points forms a local reference frame for the local neighbourhood around the point. They compute their ISS descriptor for each of the detected salient points, by dividing up a spherical support region around the point into bins, and counting the number of points that fall into each bin, producing a signature descriptor. The local reference frame is used to determine the coordinate of the bin in the histogram. The authors note that the local reference frame computed using PCA analysis has sign ambiguities that can result in 4 different local reference frames if the normal direction is not known, and 2 if the normal direction is known. Therefore, the authors compute 4 different version of their descriptor for each point to ensure all possibilities are captured. They achieve 98% average recognition for the problem of finding a single car object in a scene. This is done for 72 different car models of similar shape for two different scenes. The authors do not report the occlusion ratio.

Tombari *et al.* [33] propose a technique that discusses the repeatability of the invariant local reference frame that is used in the computation of many LSDs. As previously noted, the computation of the local reference frame can result in up to 4 different versions requiring 4 different versions of the descriptor. Clearly, this ambiguity slows down the matching stage significantly. Another problem with local reference frame computation is that they are sensitive to occlusion, clutter, and noise in the scene due to the corruption of the support region. Since the computation of the LSD depends on the local reference frame, the LSDs of the corresponding points might not match. In order to compute a local reference frame, the principle components of the local neighbourhood are computed. To compute the principle components, eigenvalue decomposition is performed on the covariance matrix, $COV(p)$, of the k points in $N_r(p)$ defined as:

$$COV(p) = \frac{1}{k}\sum_{i=1}^{k}(p_i - \hat{p})(p_i - \hat{p})^T, \quad \hat{p} = \frac{1}{k}\sum_{i=1}^{k} p_i \tag{6}$$

The authors propose a weighted version of the covariance matrix where points closer to the support contribute more than points further away from the support. This essentially means that if part of the neighbourhood is corrupoted with occlusion, clutter or noise, then this corruption will have less of an effect on the eigenvalue calculation due to the fact that it will likely be in the outer portion of the support region. The new covariance matrix for support region of radius, r, is defined as:

$$COV(p) = \frac{1}{\sum_{i:d_i \leq r}(r - d_i)} \sum_{i:d_i \leq r} (r - d_i)(p_i - p)(p_i - p)^T \tag{7}$$

where $d_i = ||p_i - p||_2$ and p replaces $\hat{p}$ for the sake of efficiency. In order to disambiguate the sign of the principle components each of the principle components are simply reoriented in the direction of the majority of the vectors it is representing [34].

In formulating their descriptor, the authors take inspiration from well established and successful 2D descriptors such as SIFT [35] and SURF [36]. They stress that part of the reason for the success of these descriptors for 2D object recognition is the use of sub-local histograms that describe a specific region within the support region. Additionally, these descriptors use first order differential entities (gradients in 2D and normals in 3D), which are highly descriptive. Therefore, the authors construct their descriptor by first dividing the spherical support region around the support, p, into a spherical grid based on polar coordinates. Next, for each grid cell a local histogram which bins the angular differences between $\mathbf{n}_p$, and points within the grid cell is built. These sub-local histograms are concatenated into a single histogram to create their descriptor. Their descriptor is dubbed *Signature of Histograms of Orientations (SHOT)*. The authors also discuss a spatially varying binning strategy where normals whose angular difference to $\mathbf{n}_p$ is small are placed in wider bins while those with a bigger difference are placed into increasingly finer bins. The intuition for this binning strategy is that points whose orientation differ significantly from the orientation of the support are the most descriptive, and therefore should be more highlighted in the descriptor. The authors compare their technique to Spin Images, Normal Field. They show that their method outperforms the others in terms of precision and recall but they do not report actual recognition rates. The authors extend their work in [37] by adding colour information to the descriptor. They simply augment the sub-local histograms by binning the difference between the colour vector for each point in a cell and the colour vector of the support. They found that this improved their results.

Taati and Greenspan [38] present a new approach to the construction of local shape descriptors by enumerating most geometric properties that are encoded in an LSD as proposed in previous literature. They coin the term *Variable Dimensional Local Shape Descriptor (VD-LSD)* to describe their descriptor. They categorize the properties encoded between a reference point and a point in its neighbourhood into three types: position scalars, direction scalars, and dispersion scalars. Instead of trying to formulate a single universal LSD that performs well on all objects, the authors' premise is that certain properties might be better in describing certain surface geometries than others. The goal then becomes to find the best subset of properties to encode in the LSD for a specific object. The authors enumerate 25 different properties and evaluate the use of heuristic algorithms such as Simulated Annealing and Genetic Algorithms [39] to find the best subset out of the 25. The authors also show that some of these properties are linearly dependent and therefore exclude some of them in their selection process. Once the best subset of properties is chosen the LSD is constructed using a multi-dimensional histogram which uses a dimension for each of the chosen properties. In order to recognize an object in the scene, the optimal set of properties for that object was used to compute the scene LSDs and then matching using RANSAC was performed. The authors show that their VD-LSD descriptors achieve a recognition rate of 83.8% versus 53.8% for Spin Images on an augmented Standard 3D Object Recognition Benchmark dataset in which they supplemented the standard benchmark with 6 more models and 636 new scenes. As expected the increased number of RANSAC iterations lead to a higher recognition rate for all methods

3.1.2 Point Pair Descriptors

In [40], [41], the *Point Pair Feature (PPF)*, which is a local shape descriptor which describes geometric properties associated with two oriented points instead of a local neighbourhood around a point is proposed. Both works highlight some of the weaknesses of local shape descriptors which describe a local neighbourhood and they include:

- Sensitivity to occlusion, local clutter, noise and low sampling resolution.
- Inability to discriminate self-similar objects parts such as planar patches.

Essentially, these weaknesses cause the LSDs of local neighbourhoods to be less *repeatable* and less discriminating, where repeatability is a measure of how likely it is to find the LSD in both the model and the scene. Given two oriented points, (p, q), with normals, $(\mathbf{n}_p, \mathbf{n}_q)$, and a vector connecting the two points, $\mathbf{d}$, the Point Pair Feature is defined as:

$$\mathbf{F}(p,q) = (||\mathbf{d}||_{\mathbf{2}}, \angle(\mathbf{n}_p, \mathbf{d}), \angle(\mathbf{n}_q, \mathbf{d}), \angle(\mathbf{n}_p, \mathbf{n}_q)) \tag{8}$$

In [41] a global model description is created for each model in the database by sampling point pairs at a specific distance from each other and whose normals differ by a specific angle on the model. The distances and the angles are sampled in steps of d_{dist} and d_{angle}. For each model point pair,(m_i, m_j), $\mathbf{F}(m_i, m_j)$ is calculated and the pair (m_i, m_j) is placed into a 4D hash table whose key is the value of $\mathbf{F}(m_i, m_j)$. This 4D hash table represents the global description of the model.

During the online recognition phase a reference scene point, s_r is sampled and paired with another scene point s_i. Then, $\mathbf{F}(s_r, s_i)$ is calculated and the 4D hash table of the model is used to find corresponding model pairs. For each model pair, (m_r, m_i), where m_r is selected to be the model reference point, s_r is registered to m_r. The registration of s_r to m_r includes the registration of their normals as well. This registration constrains 5 degrees of freedom of the 6D pose but leaves one free. In order to complete the registration of the pairs, the angle, α which aligns s_i to m_i is calculated. A vote is cast for the coordinate (m_r, α) in a 2D accumulator array. A different s_i is chosen from the scene and the operation is repeated until all scene points have been paired with the reference scene point s_r. The maximum peaks of the accumulator array are taken as candidate local coordinates (m_r, α) that would align the model to the scene. Since a reference point s_r might not lie on the model the process is repeated for different references points from the scene. The poses generated by the voting scheme of Point Pair Feature matching are clustered together such that poses within a cluster do not differ in rotation and translation by more than predefined thresholds. The score of the cluster which is the sum of the scores of the poses within the cluster is calculated. The score of the pose is the number of votes that the pose received. All clusters with low scores are removed. For all other clusters, the poses are averaged to generate a more accurate pose. The authors report a recognition rate of 97% on the Standard 3D Object Recognition Benchmark using $|S|/5$ reference points during online matching for scenes containing an occlusion rate of up to 84%. This is slightly better than the results reported by the Tensors technique. The method also performed object recognition 5 seconds faster per object as compared to the Tensors technique. The authors also report results for detection of an object in single object scenes; scenes with zero clutter and only self occlusion. Noisy single object scenes are generated by corrupting the 3D point positions with various levels of Gaussian noise. The standard deviation of the Gaussian noise distribution is set to a percentage of the object diameter. The technique achieved around 81% detection rate at a standard deviation set to 4% of the model diameter for the Gaussian noise distribution.

Papazof *et al.* [40] proposes a similar technique that investigates using the Point Pair Feature. The offline recognition phase is very similar to that of [41] where models are represented using a

4D hash table based on the Point Pair Feature. However, in this case all model pairs are hashed into the same hash table and hash table entries that contain too many entries are deleted. This serves to keep only discriminating Point Pair Features while removing non-discriminating ones. In the online recognition phase, a RANSAC framework is proposed where every hypothesis resulting from point pair correspondence is evaluated using an acceptance function. All hypotheses that satisfy the acceptance function are further refined by removing conflicting hypotheses. The details of hypothesis verification are explained in the next section. They also report results that outperform the Tensors and Spin Image technique and in fact achieve a recognition rate of 100% for objects with less than 84% occlusion in 48 out of the 50 test scenes and around 97% for the remaining two scenes. As in [41], they also show the robustness of their technique to various levels of noise. At a noise level set to 4% of the model diameter the authors report a recognition rate around 65% in multi-object scenes.

The work on the *Point Feature Histogram(PFH)* and extensions of it proposed by Radu *et al.* [42], [43], [44] is used for the end goal of object class recognition in scenes with low clutter and occlusion. The PFH LSD combines the idea of describing a local neighbourhood as in Spin Images, and the idea of describing point pairs as in the Point Pair Feature. Their LSD is formulated by calculating the Point Pair Feature for all pairs of points within a local neighbourhood and binning the values of the point pair feature into a 4 dimensional histogram. The descriptor has a computation cost of $\mathrm{O}\left(k^2\right)$ where k is the number of points in the local neighbourhood. The authors simplify their descriptor [43], and propose the *Fast Point Feature Histogram(FPFH)* LSD by first calculating the *Simplified Point Feature Histogram(SPFH)* for every point in the 3D cloud. The SPFH is computed by calculating the Point Pair Feature between the point and all it's k neighbours instead of doing so for every pair of points in the neighbourhood. In the next stage, the SPFH of every point is weighted by the SPFH of neighbouring points to create the FPFH point descriptor. The computation of FPFH takes only $\mathrm{O}\left(k\right)$ as opposed to $\mathrm{O}\left(k^2\right)$. Both of PFH and FPFH can be used as LSDs for 3D object recognition but the authors use them to segment objects into geometric primitives such as planes, sphere, edges, and corners. The segmentation of objects into geometric primitives is used to create a global shape descriptor called the *Global Point Feature Histogram(GPFH)* [44]. The GPFH descriptor is then used to classify objects into functional classes using a Support Vector Machine (SVM) classifier. The scenes dealt with contain low clutter and occlusion for the application of robotic manipulation of kitchenware. FPFH is first used to segment each object into geometric primitives. Next, a single GPFH descriptor is calculated for the entire object. This is done by first voxelizing the object using an octree and casting a ray, r_{ij}, between every pair of voxels. For every voxel that r_{ij} passes through, the dominant geometric primitive is determined. The dominant geometric primitive is the one which has the largest representation within the voxel. Next, For every consecutive pair of voxels the transition from the dominant primitive of the first voxel to the next is recorded. These transitions are then binned into a histogram of all possible transitions creating the GPFH. An SVM classifier uses the GPFH to classify the objects into different classes. The aurhors acheived 98.7% accuracy for primitive shape segmentation and 96.9% accuracy for functional object class recognition. Although this is a global shape descriptor, the idea of using a local shape descriptor to build a global one is interesting because it describes the object at different levels of locality.

3.1.3 Local Patch Descriptors

Lam and Greenspan[18] also point out the same issues that affect the repeatability of LSDs based on local neighbourhoods. Instead of calculating LSDs for local support regions, the authors support the hypothesis that interest points are repeatable under conditions such as different sampling resolution, and sensor noise. Therefore they propose an object recognition technique that is based on interest points without calculating any LSDs. Their technique still has local characteristics and that is the reason it is reviewed in this section. The technique can be described using the following steps:

i *Interest Point Extraction*: The difference of normals operator [45] is used to extract points of interest. The operator calculates the normal of a point based on two differently sized neighbourhoods; a small and a large. Points that lie on areas of high curvature will have a significant difference in their calculated normals and such points are considered interest points.

ii *Boundary Curve Reconstruction*: The interest points are joined into closed curves called interest curves.

iii *Region Segmentation*: Based on the boundary curves, the points are segmented into labeled regions called *interest segments* which are used for matching.

In order to perform the actual matching the 4 Congruent Point Sets (4PCS) [11] algorithm is used to locate a matching interest segment in the scene. 4PCS is a registration technique that does not require the prior calculation of any interest points or local shape descriptors and therefore it can be used to register raw point segments. In order to compare how well the interest segments in the database were segmented in the scene, the authors develop a measure to quantify the quality of the segmentation. Experiments showed that the interest segments are repeatable and therefore effective for object recognition. It is also important to note that a single matching segment is enough to register the object to the scene. In [1], the authors improve the quality of their segments by including a supervised merging stage to merge over-segmented neighbouring regions which increases the repeatability of the segments as well as increasing the match between corresponding segments. In their most recent work [46], the authors propose an object recognition system with an unsupervised merging stage and achieve a 93% recognition rate on a high resolution LiDAR dataset and 81% on a noisy and low resolution Kinect dataset. Occlusion rates are not reported for their datasets. Online efficiency of the method is an issue due to the fact that 4PCS is a quadratic complexity algorithm for registering two point sets.

3.2 Hypothesis Verification

Once LSDs are computed for the scene they are compared against the LSDs in the model database and correspondences are established between 3D points whose LDSs are similar according to a similarity measure. In the case that the LSD includes the local reference frame such as the SHOT descriptor [33], only one correspondence is necessary to find the transformation from model to scene. In the case that the LSD does not include the local reference frame, three correspondences are required to find a transformation. These correspondences generate a set of hypothesis, H, where an element in H, $h_i = \{M_i, T_i\}$, represents the possibility of the existence of M_i with pose T_i in the scene. Each hypothesis, $h_i = \{M_i, T_i\}$, is verified by applying the transformation T_i to the model M_i and computing the proportion of model points explained by the transformation. In other words, the proportion of model points that have a corresponding scene point after the transformation is applied. If it is above a specified threshold then the hypothesis is accepted and refined using ICP.

Most techniques apply the simple verification process outlined above. However, there are a few techniques that explicitly address the *Hypothesis Verification (HV)* stage [3], [40], [31], and [28], although none to nearly the same extend as the work of Aldoma *et al.* [28]. As indicated in [28] the HV stage of the object recognition pipeline has been relatively unexplored so far. In this section, different HV techniques proposed in the literature are reviewed.

In the Tensors technique [3] the verification procedure first verifies that the simplified scene mesh, $\mathbf{M}'_S$, and a simplified model mesh, $\mathbf{M}'_M$, are a good match and then proceeds to verify that the non-simplified versions of the meshes $\mathbf{M}_S$ and $\mathbf{M}_M$, are a good match. The first step in the verification procedure is to transform $\mathbf{M}'_M$ to the scene and calculate the following quantities:

$$\alpha = \frac{\text{corresponding vertices of model with } M'_S}{\text{total vertices of model}} \tag{9}$$

$$f_1 = \alpha D \tag{10}$$

In (9) vertices are considered corresponding if the distance between them is less than twice the mesh resolution. In (10), D is the similarity between the model and scene Tensor and α measures the quality of alignment, or in other words, the proportion of model points explained by the transformation. Furthermore, f_1 is a confidence measure which is proportional to the similarity and how well the objects are aligned. A hypothesis is allowed to proceed to the next verification step only if these two quantities are above minimum thresholds. Setting a low threshold for α allows for wrong alignments to pass through while setting a high threshold for it could exclude correct alignments with objects that are highly occluded. Therefore two thresholds, $t_{\alpha 1}$ and $t_{\alpha 2}$, are used. If $\alpha < t_{\alpha 1}$ the hypothesis is rejected. If $\alpha > t_{\alpha 2}$ the hypothesis is accepted. Otherwise if $t_{\alpha 1} \leq \alpha \leq t_{\alpha 2}$, then the hypothesis is only allowed to proceed if $f_1 > t_{f1}$. The authors experimentally determine these thresholds. In the next stage, $\mathbf{M}_M$ is transformed to $\mathbf{M}_S$ and the transformation is refined using ICP. Once this is accomplished, all non-visible model points are removed and the alignment using equation 9 is recalculated as the quantity α_2. If α_2 is higher than 0.8 the hypothesis passes through to the final stage where violations of the active space of the sensor by the transformed model are checked. An example of an active space violation is a portion of the model laying in between the sensor and detected scene points in the z-direction.

In [40], the hypotheses that are generated during the matching stage are evaluated using an acceptance function that is composed of a support term and a penalty term. The support term is the quality of alignment as defined in [3] and the penalty term is the number of model points that occlude other scene points. A hypothesis passes on to the next verification stage if the support is higher than a predefined threshold and the penalty is lower than a predefined threshold. In the final stage, conflicting hypotheses are removed, two hypotheses are conflicting if their transformed models align with a shared set of points. Conflicting hypotheses are placed in a graph where each node represents a hypothesis and an edge connects two hypotheses if they are conflicting. Non-maximum suppression is performed whereby a node is removed if it has a better neighbouring node; in other words, if the neighbouring hypothesis has a larger support term.

Aldoma *et al.* [28] focuses only on the hypothesis verification stage and present an elaborate hypothesis verification technique. Instead of treating each hypothesis individually and applying thresholds that depend on geometric cues (constraints) the authors consider the entire set of hypotheses as a global scene model. They propose a global cost function that measures the quality of the entire set of hypotheses and find the subset of hypotheses that minimizes the global cost function. The advantage of their approach is that they take interaction between different hypothesis into account which is ignored in other hypothesis verification techniques except [40] which uses a conflict graph to rule out conflicting hypotheses. In addition, their approach brings a significant reduction to the number of hard thresholds that are generally used in other hypothesis verification techniques. Formally, given the set of all hypotheses, $H = \{h_1, \ldots, h_n\}$, a solution is $X = \{x_1, \ldots, x_n\}$ having the same cardinality as H with $x_i \in \mathbb{B} = \{0, 1\}$ indicating whether the corresponding hypothesis $h_i \in H$ is dismissed/included. Therefore the solution space has a cardinality of 2^n. Given a global cost function $\mathfrak{F}(X) : \mathbb{B}^n \to \mathbb{R}$ the goal is to find the solution that minimizes $\mathfrak{F}(X)$. Since the solution space is large, the authors use simulated annealing to find an approximate optimal solution. Let M_{h_i} be the model after being transformed to the scene using the associated transformation, T_i. In addition, model points that are occluded by scene points after the transformation are removed from M_{h_i}. Given this definition, the global cost function is composed of four geometric cues that either penalize or support a hypothesis:

i **Scene fitting**: A weight explaining how well a scene point, s_i, is explained by M_{h_i}.

ii **Model Outliers**: Penalizes points in M_{h_i} that do not explain any scene point.

iii **Multiple Assignment**: Penalizes scene points that are explained by more than one hypothesis.

iv **Clutter**: In a realistic object recognition scenario, it is not guaranteed that all objects in the scene are part of the model database; such objects are called clutter objects. Ideally, the object recognition method should simply ignore such objects. However, local parts of clutter objects might match local parts of database objects generating false hypotheses. The second cue which deals with model outliers will penalize such hypotheses only if M_{h_i} has enough non-occluded model outliers to significantly penalize the hypothesis. To deal with clutter in cases where the second cue does not penalize the hypothesis a clutter cue is proposed. The cue penalizes hypotheses that locally explain some part of a smooth surface patch but not nearby points belonging to the same smooth surface. The idea is that points that are parts of a smooth surface tend to belong to the same object and therefore a correct hypothesis should explain the entire smooth surface. In order to detect smooth surfaces, a region growing algorithm is applied that picks random seeds in the scene and grows the seeds into larger patches as long as a continuity constraint is satisfied. The continuity constraint is assessed by density of points in space and smoothness through surface normals. For a random scene point p_i, the continuity constraint is satisfied with regards to a neighbouring point, p_j, if p_j distance to p_i is below a threshold, t_d, and the angular difference between their normals is also below a threshold, t_n.

In order to demonstrate the effectiveness of their hypothesis verification technique the authors implement an object recognition system that uses the SHOT descriptor [33] to generate the hypotheses set and then apply their hypothesis verification technique. The authors compare their results with state of the art recognition techniques of Papazof *et al.*, Drost *et al.*, and Bariya *et al.* They outperform all methods on the standard object recognition benchmark and in fact achieve a recognition rate of 100% for all scenes in the dataset which include a maximum occlusion rate of 91.4%.

3.3 Comparison of Local Techniques

We compare the techniques proposed so far in two ways. The first comparison looks at the different properties of the LSDs and is summarized in Table 1. The second comparison compares the recognition rate of all the techniques that use the Standard Object Recognition Benchmark and this is shown in Table 2.

LSD	Describes	Dimensionality	Local Frame
Spin Images [2]	Local Neighbourhood	2	No
3D Shape Context [26]	Local Neighbourhood	3	Ambiguous
Tensors [3]	Local Neighbourhood	3	Ambiguous
ISS [32]	Local Neighbourhood	3	Ambiguous
Normal Field [31]	Local Neighbourhood	3	Ambiguous
SHOT [33]	Local Neighbourhood	3	Unique
VDLSD [38]	Local Neighbourhood	up to 9	No
PPF [40], [41]	Point Pairs	4	Ambiguous
PFH, FPFH [42], [43]	Local Neighbourhood & Point Pair	4	No
Repeatable Segments [46]	3D point segment	n/a	No

Table 1: A comparison of local shape descriptor construction criteria

LSD	Average Recognition Rate
Spin Images [2]	87.8%
Tensors [3]	96.5%
Normal Field [31]	97.5%
PPF [41]	97%
PPF [40]	99.9%
Hypothesis Verification with SHOT [28]	100% [1]

[1] This recognition rate is for the maximum occlusion rate of the dataset of 91.4%. All other results are for scenes with up to 84% occlusion only.

Table 2: A comparison of LSD performance on the Standard 3D Object Recognition Benchmark for objects in the scene with up to 84% occlusion.

4 Future Research Directions

Although Local Shape Descriptors have shown great promise in solving the correspondence problem there is still room for improvement on many fronts and they include:

- *Dealing with Harder Datasets*: Although very good results were reported for the Standard 3D Object Recognition Benchmark, this dataset only contains 5 objects with 50 scenes. For example, Spin Image's performance dropped from 87.8% to 53.8% on the larger dataset used by Taati and Greenspan [38]. This is an indication that as the database gets larger the LSDs become less discriminating. Furthermore, the Standard 3D Object Recognition Benchmark is also a high quality dataset whereas lower quality sensors such as the Microsoft Kinect are generally what is used in many of today's applications. Therefore, the LSDs need to achieve high recognition rates at higher levels of noise.

- *Hypothesis Verification*: As highlighted by Aldoma *et al.* [28] the hypothesis verification stage of the object recognition pipeline has not received much attention in the literature and they have shown that a good hypothesis verification technique can significantly improve the recognition rate. Therefore there is an opportunity for new hypothesis verification techniques.

- *Scene Segmentation*: Segmenting complex free-form objects in a cluttered scene is a difficult problem which itself deserves attention. However, although accurate object segmentation might not be possible, segmentation could still be exploited to improve object recognition as shown in [28] and [46]. Both of these techniques do not attempt to fully segment the object out of the scene, but instead segment either the scene and or the models into smooth surface patches. Such segmentation was used in [28] as a clue that points on the same surface patch must be part of the same object. Such a clue is a powerful tool in dealing with clutter. Further exploitation of segmentation combined with the power of LSDs can produce techniques that are more powerful than techniques that only use LSDs.

- *Multiple Point Features*: It was shown in [40] and [41] that the Point Pair Feature is a powerful feature that competed with LSDs based on local neighbourhoods around a point. It remains to be seen if this idea can be extended further to multiple point features with more than two points. Additionally, it is possible that there is a more descriptive Point Pair Feature than the one proposed in the literature.

- *Intensity and Colour*: Most laser scanners not only give information about the 3D coordinate of the point but also the laser beam intensity of the point. Analogously, RGB-D sensors such as the Microsoft Kinect return colour information instead of laser beam intensity. Further research needs to be done into exploiting this extra information for 3D object recognition.

- *LSD Optimization*: The authors of VD-LSD proposed the idea of finding the optimal LSD to describe each object. This idea could be further extended to find the optimal LSD to describe a particular region or segment of the object. For example a different LSD would be used to describe the face of a model than the one to describe its body.

- *Weighing LSD Uniqueness*: Not all LSDs have the same discriminative power. It could be the case that a single LSD for a particular local neighbourhood of a model is different than all other LSDs. Therefore matching this single LSD in the scene implies that that object exists in the scene no matter how occluded that object is. This would eliminate the need to set a minimum threshold on the proportion of the model that must exist in the scene in order for it to be recognized.

- *Automatic Support Distance*: The support distance or angle used to calculate a LSD is manually chosen in all techniques except for [31]. Techniques to automatically select the support distance to optimize recognition have not been thoroughly explored.

- *Hierarchical Object Description*: Another idea that is sometimes explored in the literature is that of a hierarchical description of the model [47]. Instead of describing an object at one local level LSDs that are closer together could be combined to create a less local but more descriptive descriptor. Applying this idea multiple times can create a multi-level description of a single object. Recognition would then be attempted at the most global level and go down to the most local.

- *Using 2D techniques for 3D Object Recognition*: In recent works such as [47] which is not reviewed in this report a 2D technique is extended to deal with 3D data by simply treating the depth value (or the z-coordinate) of the 3D data as an intensity value of a 2D pixel. In fact, the Microsoft Kinect's data is an RGB images with a depth value associated with each pixel. The work of 2D and 3D Object Recognition has been happening in separation for the most part up to this point. However, new data types might be bringing the two problems closer together. Therefore, it might be possible to combine techniques from both areas to achieve better results.

Clearly, the problem of 3D Object Recognition is far from solved and high recognition rates are still difficult to obtain on complex datasets. As outlined above, there are many possible directions that need to be explored to try and improve recognition rates.

References

[1] J. Lam and M. Greenspan, "Shape matching of repeatable interest segments in 3d point clouds," in *IEEE Conf. Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 25–32, 2012.

[2] A. E. Johnson and M. Hebert, "Surface matching for object recognition in complex three-dimensional scenes," *Image and Vision Computing*, vol. 16, no. 9, pp. 635–651, 1998.

[3] A. Mian, M. Bennamoun, and R. Owens, "Three-dimensional model-based object recognition and segmentation in cluttered scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 10, pp. 1584–1601, 2006.

[4] S. Ullman and G. J. Power, "High-level vision: Object-recognition and visual cognition," *Optical Engineering*, vol. 36, no. 11, pp. 3224–3224, 1997.

[5] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, "Matching 3d models with shape distributions," in *IEEE Int. Conf. Shape Modeling and Applications*, pp. 154–166, 2001.

[6] E. Wahl, U. Hillenbrand, and G. Hirzinger, "Surflet-pair-relation histograms: a statistical 3d-shape representation for rapid classification," in *IEEE* 4^{th} *Int. Conf. 3-D Digital Imaging and Modeling*, pp. 474–481, 2003.

[7] G. Hetzel, B. Leibe, P. Levi, and B. Schiele, "3d object recognition from range images using local feature histograms," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. II–394, 2001.

[8] L. Shang and M. Greenspan, "Real-time object recognition in sparse range images using error surface embedding," *Int. J. Comput. Vision*, vol. 89, no. 2-3, pp. 211–228, 2010.

[9] P. Besl and N. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992.

[10] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[11] D. Aiger, N. J. Mitra, and D. Cohen-Or, "4-points congruent sets for robust surface registration," *ACM Trans. Graph.*, vol. 27, no. 3, pp. #85, 1–10, 2008.

[12] D. H. Ballard, "Generalizing the hough transform to detect arbitrary shapes," *Pattern recognition*, vol. 13, no. 2, pp. 111–122, 1981.

[13] K. Khoshelham, "Extending generalized hough transform to detect 3d objects in laser range data," in *Proc. ISPRS Workshop on Laser Scanning*, pp. 206–210, 2007.

[14] W. E. L. Grimson and T. Lozano-Perez, "Model-based recognition and localization from sparse range or tactile data," *The Int. J. of Robotics Research*, vol. 3, no. 3, pp. 3–35, 1984.

[15] W. E. L. Grimson and T. Lozano-Perez, "Localizing overlapping parts by searching the interpretation tree," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 4, pp. 469–482, 1987.

[16] M. Greenspan, "The sample tree: A sequential hypothesis testing approach to 3d object recognition," in *IEEE Conf. Computer Vision and Pattern Recognition(CVPR)*, pp. 772–779, 1998.

[17] F. Tombari, S. Salti, and L. Di Stefano, "Performance evaluation of 3d keypoint detectors," *Int. J. Comput. Vision*, pp. 1–23, 2012.

[18] J. Lam and M. Greenspan, "On the repeatability of 3d point cloud segmentation based on interest points," in *IEEE 9^{th} Conf. Computer and Robot Vision*, pp. 9–16, 2012.

[19] C. S. Chua and R. Jarvis, "3d free-form surface registration and object recognition," *Int. J. Comput. Vision*, vol. 17, no. 1, pp. 77–99, 1996.

[20] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *JOSA A*, vol. 4, no. 4, pp. 629–642, 1987.

[21] A. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 5, pp. 433–449, 1999.

[22] H. Murase and S. K. Nayar, "Visual learning and recognition of 3-d objects from appearance," *Int. J. Comput. Vision*, vol. 14, no. 1, pp. 5–24, 1995.

[23] S. Ruiz-Correa, L. G. Shapiro, and M. Melia, "A new signature-based method for efficient 3-d object recognition," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. I–769, 2001.

[24] W. B. Johnson and J. Lindenstrauss, “Extensions of lipschitz mappings into a hilbert space,” *Contemporary Math.*, vol. 26, no. 189-206, p. 1, 1984.

[25] D. Fradkin and D. Madigan, “Experiments with random projections for machine learning,” in *ACM* 9th *Int. Conf. Knowledge Discovery and Data mining (SIGKDD)*, pp. 517–522, 2003.

[26] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik, “Recognizing objects in range data using regional point descriptors,” 2004.

[27] J. Foley, A. Van Dam, S. Feiner, J. Hughes, and R. Phillips, *Introduction to Computer Graphics.* Addison-Wesley, 1994.

[28] A. Aldoma, F. Tombari, L. Di Stefano, and M. Vincze, “A global hypotheses verification method for 3d object recognition,” in *European Conf. Computer Vision (ECCV)*, pp. 511–524, 2012.

[29] J. Novatnack and K. Nishino, “Scale-dependent 3d geometric features,” in *IEEE* 11th *Int. Conf. Computer Vision*, pp. 1–8, 2007.

[30] J. Novatnack and K. Nishino, “Scale-dependent/invariant local 3d shape descriptors for fully automatic registration of multiple sets of range images,” in 10th *European Conf. Computer Vision*, pp. 440–453, 2008.

[31] P. Bariya and K. Nishino, “Scale-hierarchical 3d object recognition in cluttered scenes,” in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 1657–1664, 2010.

[32] Y. Zhong, “Intrinsic shape signatures: A shape descriptor for 3d object recognition,” in *IEEE* 12th *Int. Conf. Computer Vision Workshops (ICCV Workshops)*, pp. 689–696, 2009.

[33] F. Tombari, S. Salti, and L. Di Stefano, “Unique signatures of histograms for local surface description,” in *European Conf. Computer Vision (ECCV)*, pp. 356–369, Springer, 2010.

[34] R. Bro, E. Acar, and T. G. Kolda, “Resolving the sign ambiguity in the singular value decomposition,” *J. Chemometrics*, vol. 22, no. 2, pp. 135–140, 2008.

[35] D. G. Lowe, “Object recognition from local scale-invariant features,” in *IEEE* 7th *Int. Conf. Computer vision*, vol. 2, pp. 1150–1157, 1999.

[36] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (surf),” *Comput. Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[37] F. Tombari, S. Salti, and L. Di Stefano, “A combined texture-shape descriptor for enhanced 3d feature matching,” in *IEEE* 18th *Int. Conf. Image Processing (ICIP)*, pp. 809–812, 2011.

[38] B. Taati and M. Greenspan, “Local shape descriptor selection for object recognition in range data,” *Comput. Vision and Image Understanding*, vol. 115, no. 5, pp. 681–694, 2011.

[39] D. Lawrence, *Genetic Algorithms and Simulated Annealing.* Los Altos, CA: Morgan Kaufman Publishers, Inc., 1987.

[40] C. Papazov and D. Burschka, "An efficient ransac for 3d object recognition in noisy and occluded scenes," in *Asian Conf. Computer Vision (ACCV)*, pp. 135–148, Springer, 2011.

[41] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3d object recognition," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 998–1005, 2010.

[42] R. B. Rusu, Z. C. Marton, N. Blodow, and M. Beetz, "Persistent point feature histograms for 3d point clouds," 10^{th} *Int. Conf. Intelligent Autonomous Systems (IAS)*, p. 119, 2008.

[43] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *IEEE Int. Conf. Robotics and Automation (ICRA)*, pp. 3212–3217, 2009.

[44] R. B. Rusu, A. Holzbach, M. Beetz, and G. Bradski, "Detecting and segmenting objects for mobile manipulation," in *IEEE* 12^{th} *Int. Conf. Computer Vision Workshops (ICCV Workshops)*, pp. 47–54, 2009.

[45] Y. Ioannou, *Automatic urban modelling using mobile urban LIDAR data.* PhD thesis, Queen's University, 2011.

[46] J. Lam and M. Greenspan, "3d object recognition by surface registration of interest segments," in *IEEE Int. Conf. 3D Vision*, 2013 in press.

[47] L. Bo, K. Lai, X. Ren, and D. Fox, "Object recognition with hierarchical kernel descriptors," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 1729–1736, 2011.