COMPRESSED VFH DESCRIPTOR FOR 3D OBJECT CLASSIFICATION

Yasir Salih1,2,3, A.S. Malik2, D. Sidibé3, M.T. Simsim4, N. Saad2 and F. Meriaudeau3

1Science and Technology Unit, Umm Al-Qura University, 21955 Makkah, Saudi Arabia
2Centre for Intelligent Signal and Imaging Research, Universiti Teknologi Petronas, 31750, Tronoh, Malaysia
3Le2i laboratory, UMR, CNRS 6306, Université de Bourgogne, 71200 Le Creusot, France
4Electrical Engineering Department, Umm Al-Qura University, 21955 Makkah, Saudi Arabia

ABSTRACT

This paper presented a compressed version of viewpoint feature histogram descriptor (VFH) for object classification based on shape recognition. VFH is known for representing geometrical features (3D rotation angles) of 3D points with a concatenated histogram. However, this histogram is large and sparse. The proposed descriptor employs eigenvalue decomposition to extract dominant orientation features from the point cloud and using it as a descriptor instead of sparse histogram. Using dominant features maintains the main properties of the object with minimal descriptor length which is applicable for recognizing the geometrical class of 3D objects (sphere, rectangle, … etc). This descriptor was tested for two class and multi-class 3D object classification using support vector machines (SVM). The new descriptor showed promising matching results of 88% for two class classification and 83% for multiple class classification of 3D objects.

Index Terms — viewpoint feature histogram, support vector machine, point clouds, 3D descriptors

1. INTRODUCTION

Recognizing the content of a digital image rely on the ability to uniquely describe its content with a robust to noise and invariant to transformation description. This description studies the local and global properties of the image captured and puts these features in a suitable representation scheme. 3D descriptors describe the surface of the desired object [1]. Given an object represented by a set of 3D points, these descriptors assign a unique signature to the 3D object which is later used in recognition and matching applications. These descriptors describe the distribution and density of the point cloud on the surface and/or the variation in surface normal between these points [2].

Existing 3D points descriptors are mostly based on orientation gradient because it captures the internal details of the point cloud and it invariant to size, scale and rotation variations. Gradient orientation descriptors are represented as histogram that records the occurrence of each unique feature [3]. Depending on the nature of the descriptor, these histograms might encode angles, distances or captures the presence of certain shapes in the point cloud (such as circles, arcs or lines). There are also classes of 3D descriptors that integrate both shape and color information by they have the advantages of both types of features [4]. The major limitation of histogram representation is that, the histogram has a fixed grid distribution that is not adaptable to the nature of the underlying point cloud and it is mostly large in size and very sparse.

This work presents a variant of point cloud descriptor that uses eigenvalue decomposition (EVD). VFH descriptor represents geometrical properties of the underlying point cloud by encoding the orientation difference of each 3D point in the point cloud with respect to the centroid point and also encoding the normalized distance between the said points and the centroid. The descriptor is computed in the similar pattern to the viewpoint feature histogram (VFH) descriptor [5]. However, instead of using histogram to represent these differences, EVD is used to extract the dominant orientation features and use it to represent the descriptor. Since EVD extracts only dominant features, thus the descriptor will be more informative and less sparse.

The remaining of this paper is organized as follows; section 2 presents previous works on point cloud descriptors and how they were employed in various applications. Section 3 presents the methodology of the proposed descriptor. Section 4 presents detailed experimental analysis on the proposed descriptor and finally Section 5 concludes the paper with the main findings.

2. RELATED WORKS

3D descriptors can be categorized into two main groups; global descriptors and local descriptors. Global descriptors describe the global geometry of a cloud of 3D points while local descriptors describe the local neighborhood around each of the given key-points. Figure 1 illustrates the differences between local and global descriptors for multiple views matching scenario [6].

![Figure 1. Global vs local point cloud descriptors](image)

2.1 Global Point Cloud Descriptors

This type of descriptors describes the global properties of the point cloud by considering all 3D points. Wahl et al. [7] computed four features for every point pair in the point cloud and then encodes the features of all possible point pairs in a histogram. Rusu et al. [8] further modified the work in [7] to introduce the point feature histogram (PFH) descriptor by using the pitch, yaw, and roll angles of every point pair in the point cloud encode them into a 3D histogram. PFH has high computational complexity as it considers all possible pairing the point cloud.

The PFH descriptor is rotation invariant and thus it is not suitable for pose estimation problems especially in mobile robotics. VFH descriptor had been proposed to include a representation for viewing direction for each 3D point in the point cloud in addition to the geometric properties of the 3D points [5]. This descriptor computes the cosine between normalized centroid point and the normal for each 3D point. The geometry of the point cloud is represented by variant of PFH that only pairs the point cloud by considering all 3D points. Wahl et al. [7] computed four features for every point pair in the point cloud and then encodes the features of all possible point pairs in a histogram. Rusu et al. [8] further modified the work in [7] to introduce the point feature histogram (PFH) descriptor by using the pitch, yaw, and roll angles of every point pair in the point cloud encode them into a 3D histogram. PFH has high computational complexity as it considers all possible pairing the point cloud.

The PFH descriptor is rotation invariant and thus it is not suitable for pose estimation problems especially in mobile robotics. VFH descriptor had been proposed to include a representation for viewing direction for each 3D point in the point cloud in addition to the geometric properties of the 3D points [5]. This descriptor computes the cosine between normalized centroid point and the normal for each 3D point. The geometry of the point cloud is represented by variant of PFH that only pairs the point cloud by considering all 3D points. Wahl et al. [7] computed four features for every point pair in the point cloud and then encodes the features of all possible point pairs in a histogram. Rusu et al. [8] further modified the work in [7] to introduce the point feature histogram (PFH) descriptor by using the pitch, yaw, and roll angles of every point pair in the point cloud encode them into a 3D histogram. PFH has high computational complexity as it considers all possible pairing the point cloud.

The PFH descriptor is rotation invariant and thus it is not suitable for pose estimation problems especially in mobile robotics. VFH descriptor had been proposed to include a representation for viewing direction for each 3D point in the point cloud in addition to the geometric properties of the 3D points [5]. This descriptor computes the cosine between normalized centroid point and the normal for each 3D point. The geometry of the point cloud is represented by variant of PFH that only pairs the point cloud by considering all 3D points. Wahl et al. [7] computed four features for every point pair in the point cloud and then encodes the features of all possible point pairs in a histogram. Rusu et al. [8] further modified the work in [7] to introduce the point feature histogram (PFH) descriptor by using the pitch, yaw, and roll angles of every point pair in the point cloud encode them into a 3D histogram. PFH has high computational complexity as it considers all possible pairing the point cloud.
segments and then it computes a VFH descriptor for each segment. This reduces the effect of noise and occlusion because they mostly happen in the border between planes in the 3D object. Adoma et al. [10] refined the clustered VFH descriptor by assigning a repeatable reference frame that enables 6DOF pose estimation.

2.2 Local Point Cloud Descriptors

Frome et al. [11] proposed 3D shape context descriptor as a direct extension to 2D shape context descriptors however it has high memory and computational complexities. Tombari et al. [12] modified 3D shape context by assigning a local reference frame which reduced the memory and computational requirements of the descriptor.

Tombari et al. [13] also proposed the signature of histogram of orientation descriptor (SHOT) which is built on orientation difference between the query point and each point in the support and then encoding these differences in a histogram. Recently an enhanced version of SHOT has been introduced that combines texture cues along with the point cloud geometry known as CSHOT (color SHOT) which shows better accuracy than the SHOT descriptor [4]. Salti et al. [14] presented a GPU implementation for SHOT to speed its computations. Rusu et al. presented a modified version of PFH descriptor known Fast Point Feature Histogram (FPFH) which is computed locally around a keypoint and it only pairs the query point with its neighbors. This reduces the computational complexity of PFH [15].

3. METHODOLOGY

This paper presented a compressed VFH descriptor using eigenvalue decomposition. EVD helps to reduce large matrices that contain redundant information into a concise representation that contains only the necessary information [16]. EVD has been in variety of computer vision problems such as face recognition [17] and it was also used for 2D descriptors [18].

VFH descriptor composes of orientation difference of each point cloud from the centroid and the normal viewing direction of each point formed into concatenated histogram. The proposed 3D descriptor is built by replacing the histogram formation part of VFH descriptor with eigenvalue decomposition projection step. This is done using a set of eigenvectors that corresponds to the dominant eigenvalues of the EVD decomposition. Certainly, a set of eigenvectors and eigenvalues have to be computed prior to the projection steps which is known as building the eigenspace for the descriptor as shown in figure 2.

3.1 Building Eigenspace

This is the process of training a set of eigenvectors designed to extract the principal components from a set of new features that has the same dimension as the eigenvectors. Training is performed using a large and comprehensive dataset of the same type of the data to be used during the testing process. However since point clouds have different size depending on the complexity of the object and the distance from camera, the point clouds used in this paper are sampled to extract 1500 3D points uniformly distributed over the original point cloud. This number had been chosen empirically which proved to give good recognition accuracy for both the new descriptor and VFH descriptor.

In order to build the eigenspace, let’s assume we have selected M point clouds that cover most of the types of objects to be encountered in real life. Also assume that for each point cloud N 3D points have been uniformly sampled and their corresponding surface normals were computed.

After that, for each one of these point clouds, the three vectors \((A, B, \Gamma)\) corresponding to the orientation angles and the viewpoint angles vector \((\Theta)\) are computed. Here we also add the normalized distance \(\delta\) for each point from the centroid point. \(\Delta = [\delta_1, \cdots, \delta_N] \) is the normalized distances vector. The next step is to compute the mean value for each one of these vectors across the large dataset of M point clouds. Naturally this mean the five feature vectors are firstly computed for each point cloud and then stored in a large matrix \((A \in \mathbb{R}^{N \times M})\) where each column represents the features computed for one point cloud and the mean feature vector will be computed across the entire columns, it is a vector \(\mathbf{M} = [m_1, \cdots, m_N]^T\). It has to be noted that there will be separate mean vector \(\mathbf{M}\) and feature matrix \(\mathbf{A}\) for each one of these five feature vectors extracted from the point cloud and thus five separate eigenvalue decompositions will be performed to create the full eigenspace. The mean vector \(\mathbf{M}\) is subtracted from each one of the columns of matrix \(\mathbf{A}\) as shown in (1) were \(\bar{\mathbf{A}} = [A_1, \cdots, A_M]\) is matrix whose columns are replica of mean vector \(\mathbf{M}\). EVD decomposes a covariance matrix into a set of eigenvalues and eigenvectors. The covariance matrix \((\Sigma)\) is computed using \(\mathbf{A}\) which also includes the mean subtraction step as in (1). Then the eigenvalue decomposition is performed using (2).

\[ \Sigma = (A - \bar{M})(A - \bar{M})^T \]  
\[ \Sigma = U \Lambda U^T \]  

The eigenvalues \((\Lambda = \text{diag}(\mathbb{D}))\) is a diagonal matrix of \(N\) elements that represents the energy of the decomposition components and it is sorted in a descending order. The eigenvectors \((U)\) are new basis vectors where the eigenvalue represents the energy carried by each set of eigenvectors (basis vectors). The effectiveness of this decomposition depends on the energy distribution in the eigenspace; it is desired that most of the energy falls in the first bins of the eigenvalues so that the remaining bins will have a negligible weight that could be eventually truncated.

In this paper, to build the eigenspace 5000 different point cloud objects have been selected from Washington University RGB-D dataset (Figure 3) [19]. This dataset contains more than 300 objects at different views which represent most of object encountered in daily life. Each one of these objects has been resampled with uniform distribution to extract 1500 3D points from it. Finally the eigenvalue decomposition had been applied as it was described earlier. At the end of this decomposition, the mean vector used for mean subtraction, the eigenvalues and the eigenvectors had been stored for each one of the five features which will later be used for building a new compressed point cloud descriptor.
3.1.1 Building the Compressed Descriptor
The process of building a new compressed point cloud descriptor starts by firstly resampling 1500 3D point from the point cloud to have the same size as the data used during the training stage. This uniform sampling is performed by firstly computing the centroid of the point cloud and then sorting the 3D points of the point cloud in a search tree according to their distance from the centroid point. After that, the 1500 points are selected with uniform distribution. It has to be noted that 1500 point are sufficient to cover all the details of simple and moderate complexity object.

Five feature vectors are computed for the point cloud as descriptor earlier which corresponds to three orientation angles between the centroid and each point in the point cloud, the normalized distance of each 3D point from the centroid point of the point cloud and lastly the viewing direction of each point with respect to the average viewing direction. Each one of these feature vector is mean subtracted using the corresponding mean vector computed during the training stage. The compressed descriptor is built by projecting the mean subtracted feature vector into a low dimensional space using the eigenvectors that correspond to the top r-eigenvalues. The number of these eigenvalues determines the length of the new descriptor. If the top 5 eigenvalues were selected, each feature will be represented with 5 bins and the overall descriptor 25 bins length because it is a concatenation three orientation angles, normalized distance and viewpoint angle descriptors respectively. Figure 4 shows process flow for creating a new compressed descriptor.

```
Input point cloud → Resampling → compute feature vectors → Concatenation → EVD projection → mean subtraction
```

4. EXPERIMENT AND RESULTS

VFH descriptor as well as the new compressed descriptor composes of orientation angles and viewpoint direction. The presence of the viewpoint component makes the descriptor vary with the viewing angle of the camera. Thus it is mostly used for 3D pose estimation applications. Since the main aim of this descriptor is to be used for 3D object classification, then the viewpoint component should be eliminated from the descriptor to allow different views of the same object to be recognized as the same object. For all classification experiments, the viewpoint component is eliminated from the descriptor which then contains three orientation angles and normalized distance only.

The new descriptor has been used for building a classification system using support vector machines classifier [20]. SVM is very famous for classifying nonlinearly separable feature by using the kernel tricks that maps the data into a high dimensional space that is separable. This work uses SVM with radial bases function kernel which has Gaussian function. The classification data was created by selecting 1000 point clouds for a number of 3D objects with different shape; two spherical shaped objects (apple and tomato), two cylindrical shape objects (food_can and soda_can) and two rectangular shaped objects (book and sponge). Each one of these objects contains at least 1000 point clouds and for each point cloud, the proposed descriptor was computed for variable lengths. The data had been divided into 75% for training (including cross validation) and 25% for testing. The classification results have been computed using three metrics which are precision, recall and accuracy as shown in (3-5). TP is the number of true positives, TN is the number true negatives, FP is the number of false positives and FN is the number of false negatives.

\[
\text{Precision} = \frac{TP}{TP + FP} \\
\text{Recall} = \frac{TP}{TP + FN} \\
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}
\]

4.1 Two-class Classification

Table 1 reports the results of two-class classification problem; Apple vs. Sponge classification which correspond to classifying spherical shape vs. rectangular one. The length of the descriptor is multiple of four because the viewpoint component is not included as discussed earlier. When using large number of eigenvalues to represent the descriptor, the feature space will be highly dimensional which leads to increase in false positive rate and thus the precision is reduced while the recall is almost perfect because nothing will be rejected. The precision tends to increase significantly by reducing the number of eigenvalues while the recall reduces slightly which indicates the presence of false negatives. At 5 eigenvalues the descriptor achieves a good balance between precision and recall and it score the best classification accuracy of 88.55%. Reducing the descriptor below this number reduces the performance as the number of eigenvalues is too few to give a unique descriptor about the 3D object. This experiment also indicates that using 5 eigenvalues which mean a descriptor length of 20 bins is the best length of the proposed descriptor for 3D object classification.

<table>
<thead>
<tr>
<th># Eigenvalues</th>
<th>Size</th>
<th>Precision</th>
<th>Recall</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>120</td>
<td>61.44</td>
<td>100</td>
<td>68.88</td>
</tr>
<tr>
<td>25</td>
<td>100</td>
<td>64.25</td>
<td>100</td>
<td>72.29</td>
</tr>
<tr>
<td>20</td>
<td>80</td>
<td>66.30</td>
<td>99.18</td>
<td>74.90</td>
</tr>
<tr>
<td>18</td>
<td>72</td>
<td>67.04</td>
<td>99.58</td>
<td>76.31</td>
</tr>
<tr>
<td>15</td>
<td>60</td>
<td>68.64</td>
<td>98.72</td>
<td>78.11</td>
</tr>
<tr>
<td>12</td>
<td>48</td>
<td>71.70</td>
<td>97.85</td>
<td>80.92</td>
</tr>
<tr>
<td>10</td>
<td>40</td>
<td>72.67</td>
<td>96.17</td>
<td>81.12</td>
</tr>
<tr>
<td>7</td>
<td>28</td>
<td>79.86</td>
<td>92.12</td>
<td>84.94</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>83.33</td>
<td>88.28</td>
<td>85.55</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>81.68</td>
<td>72.05</td>
<td>79.72</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>85.21</td>
<td>59.50</td>
<td>73.30</td>
</tr>
</tbody>
</table>

4.2 Multiple-class Classification

The second classification experiment presented multiple-class classification using support vector machines with one vs. all method. In this experiment apple has been selected from spherical shaped objects, soda_can from cylindrical shaped objects and Book from rectangular shape objects. The degree of dissimilarity between these object varies, for example a Book is very different in shape compared to an apple or soda_can. In contrast, soda_can and apple are slightly dissimilar because both have curve shaped surface.

<table>
<thead>
<tr>
<th>Descriptor length</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>apple vs. all</td>
<td>84.43</td>
<td>55.92</td>
<td>81.51</td>
</tr>
<tr>
<td>book vs. all</td>
<td>93.48</td>
<td>87.40</td>
<td>93.79</td>
</tr>
<tr>
<td>can vs. all</td>
<td>89.62</td>
<td>68.05</td>
<td>87.04</td>
</tr>
<tr>
<td>average</td>
<td>73.10</td>
<td>77.30</td>
<td>82.86</td>
</tr>
</tbody>
</table>
Table 2 presents the classification accuracy for the three objects mentioned earlier. The table shows the classification results for each object vs. the others as well as the average classification results. The second column shows the descriptor lengths at which the best classification accuracy was recorded. Looking at apple vs. all it scores the best accuracy at 12 eigenvalues per feature and it shows acceptable precision but very bad recall. A book has very distinct shape from both apple and soda can and thus it manage to achieve good classification accuracy at only 5 eigenvalues close to 94% accuracy. For the soda can vs. all, it showed better results than the one for apple but at larger number of eigenvalues with an accuracy of 87%. The average accuracy had been reported at 5 eigenvalues which was largely derived by the very good results of book vs. all at this descriptor length.

5. CONCLUSIONS

This paper presented a modified point cloud descriptor based on compressing the viewpoint feature histogram descriptor using eigenvalue decomposition. EVD extract dominant features from the point clouds and uses these features to represent the descriptor instead of building a large and sparse histogram unlike VFH and other point cloud descriptors. The proposed descriptor encodes orientation variations and distance between each of the 3D points in the point cloud and the centroid point. The descriptor had been tested for 3D object classification using SVM classifier with radial based function kernels. Good classification accuracy has been recorded; for two-class classification the descriptor scored 88% accuracy at only 20 bins descriptor and an average of 83% accuracy for multiple-class classification at also 20 bins. The descriptor describes the shape of the object and classifies object based on their shape profile. Thus for example an apple and an orange will most like be considered as one object because of their common shape profile. An improvement version of this descriptor can include texture properties of the object to make more distinction. However the size of the descriptor will be larger due to the new information. This descriptor can be further improved by using a sampling approach that includes the neighborhood properties of the sampled 3D points and by conducted further experiments to choose the best descriptor length in all scenarios.

6. REFERENCES