3D Descriptors for Object and Category Recognition: a Comparative Evaluation

Luís A. Alexandre

***Abstract*— Often practitioners face the issue of choosing the most adequate 3D features for their problem. As with many other problems, there isn't a "best" feature to use. Nonetheless, we can gauge the relative accuracy of available features on common datasets. That is what we propose to do in this paper: we describe the existing feature extraction algorithms in a public available point cloud library and perform a comparative evaluation on 3D point clouds, investigation both the object and category recognition performance. The main conclusions are: increasing the number of keypoints improves recognition results at the expense of size and time; since there are big differences in terms of recognition performance, size and time requirements, the descriptor has to be matched to the desired task; one should use a descriptor that takes advantage of color; the SHOTCOLOR descriptor shows a good balance between recognition accuracy and time complexity.**

I. INTRODUCTION

The appearance of cheap 3D cameras (such as the Xtion and the Kinect) has increased exponentially the interest in using depth information for solving vision tasks.

A useful resource for users of this type of sensors is the PCL library [1] which contains many algorithms that deal with point cloud data, from segmentation to recognition, from search to input/output.

A difficulty that arises when using such a library is "how to choose" which algorithm to use in a particular task, since many different approaches are usually available.

In this paper we intend to help a user making such a choice at the level of feature (or descriptor) extraction algorithms for 3D point clouds. We describe the available descriptor algorithms in version 1.6 of PCL, including a summary table with the most distinctive properties of each descriptor, and perform experiments to illustrate and evaluate their relative performance using a public available data set with 48 objects from 10 categories.

The paper is organized as follows: the next section discusses the recognition pipeline used in this paper; section 3 presents the evaluated descriptors; section 4 contains the experiments and the final sections contains the conclusions.

II. THE 3D OBJECT RECOGNITION PIPELINE

The object recognition pipeline used in this paper is presented in figure 1.

The input clouds are fed to the keypoint extraction algorithm. Since the cost of computing descriptors is usually high it makes sense to extract descriptors only on a subset of the input cloud. More precisely, one can identify interesting points that are somehow typical of the objects by using keypoint detectors. The idea is then to find descriptors only at keypoints: this can reduce substantially the computational cost and presumably does note impact negatively on the ability to discriminate the cloud contents.

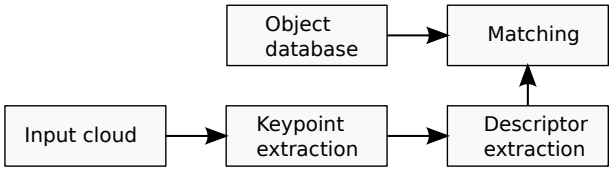

Fig. 1. Object recognition pipeline used in this paper.

In this paper we present results using two approaches for the keypoint extraction. The first uses the Harris3D keypoint extractor [2] that uses the Hessian matrix of the intensity around each point, smoothed by a Gaussian filter, to find the keypoints. The second approach consists on sub-sampling the input cloud and considering all the sub-sampled points as keypoints. The experiments made include sub-sampling with two different sizes.

After obtaining the keypoints, descriptors are obtained on the extracted keypoints and these form a set that is used to represent the input cloud. This set is matched against sets already present in the object database and the one with largest similarity (smallest distance) is considered the match for the input cloud.

Many different methods can be used to match the set of descriptors that represents a given input cloud against the available sets that represent previously registered objects (these are typically part of an object database). In this paper we will use the following set distance: find the centroid of each set plus the standard deviation for each dimension (coordinate) of each set and return the sum of the L_1 distances between them:

$$d(A,B) = L_1(c_A, c_B) + L_1(std_A, std_B)$$

where c_A and c_B are the centroids of sets A and B, and

$$std_A(i) = \sqrt{\frac{1}{|A|}\sum_{j=1}^{|A|}(a_j(i) - c_A(i))^2},\ i = 1, \ldots, n$$

and likewise for std_B; n is the size of the descriptor used in the sets. The L_1 distance between two descriptors a and

Luís A. Alexandre is with the Dept. of Informatics, Univ. Beira Interior, and Instituto de Telecomunicações, Covilhã, Portugal lfbaa at ubi.pt

TABLE I

IMPORTANT FEATURES OF THE DESCRIPTORS EVALUATED IN THIS PAPER. V=VARIABLE; Y=YES; N=NO; O=OMP; G=GPU; n=NUMBER OF POINTS IN INPUT CLOUD; m=NUMBER OF STABLE REGIONS; p=NUMBER OF AZIMUTH BINS. SEE TEXT FOR DETAILS

Descriptor	N.points	Point size	Parallel	Normals
3DSC	$n * p$	V	N	Y
CVFH	$m \leq n$	308	N	Y
ESF	1	640	N	N
FPFH	n	33	O+G	Y
PFH	n	125	G	Y
PFHRGB	n	250	G	Y
PCE	n	5	G	Y
PPF	n	5	O+G	Y
RIFT	n	32	N	N
SHOT	n	9+352	O	Y
SHOTCOLOR	n	9+1344	O	Y
USC	n	V	N	N
VHF	1	308	G	Y

b is given by:

$$L_1(a,b) = \sum_{i=1}^{n} |a(i) - b(i)|$$

III. EVALUATED DESCRIPTORS

Our goal was to evaluate the available descriptors in the current PCL version (1.6 pre-release, on June 2012).

There are some descriptors in PCL which we will not consider in this paper, since they are not applicable to point cloud data directly (need certain specific algorithms to be applied before them). These are: spin-image descriptors [3] that assume the data to be represented by a polygonal surface mesh; Global Fast Point Feature Histogram (GFPFH) [4] which assumes that the points are first labeled according to a geometric primitive class label using FPFH (see below). The Camera Roll Histogram [5] is also not considered since it is an addition to the CVFH (see below) that serves to obtain the 6DOF. Since we are only interested in evaluating the classification performance, this descriptor was left out.

Table I summarizes some important points relative to the descriptors presented in this paper. The second column contains the number of points generated by each descriptor given an input point cloud with n points. The third column shows the dimensionality of each of the generated points. The forth column indicates if there are parallelized versions of the algorithm available in the PCL (either OMP or GPU versions). The last column indicates if the algorithm requires the calculation of the surface normals at each point.

The following sub-sections present the core ideas of each descriptor. The descriptors are ordered chronologically following the year of their publication.

A. 3D Shape Context

This descriptor is was proposed in [6]. It uses a spherical grid on each of the keypoints. The north pole of the grid is oriented as the surface normal at the keypoint and the grid consists of bins along the radial, azimuth and elevation dimensions. The divisions along the radial dimension are logarithmically spaced. The number of bins can be set by the user. Each bin makes a weighted count of the number of points that fall into it. The weights used are inversely proportional to the bin volume and the local point density. Since the axes tangent to the surface are placed randomly, there is the need to extract as many versions of this descriptor per database object as there are divisions along the azimuth direction. All these versions of the descriptor need to be tried on a test cloud to find an object match.

B. Rotation Invariant Feature Transform

The RIFT descriptor [7] was developed to generalize the SIFT descriptor [8]. A circular normalized patch is built at each input point. The circular patch is divided into 4 rings of equal width. For each ring, a histogram of gradient orientations with 8 bins is computed, thus producing a 32 value descriptor for each input point. The orientations of this histogram are obtained w.r.t. the radial outward direction at each point.

C. Point Feature Histograms

Point Feature Histograms (PFH) [9] descriptor's goal is to generalize both the surface normals and the curvature estimates. Given two points, p and q, a fixed reference frame consisting of the three unit vectors (u, v, w) is built centered on p using the following procedure: 1) the vector u is the surface normal at p; 2) $v = u \times \frac{p-q}{d}$ 3) $w = u \times v$; where $d = \|p - q\|_2$. Using this reference frame, the difference between the normals at p (n_p) and q (n_q), can be represented by : 1) $\alpha = \arccos(v \cdot n_q)$; 2) $\phi = \arccos(u \cdot (p - q)/d)$; 3) $\theta = \arctan(w \cdot n_p, u \cdot n_p)$. The angles α, ϕ, θ and the distance d are computed for all pairs in the k-neighborhood of point p. In fact, usually the distance d is dropped as it changes with the viewpoint, keeping only the 3 angles. These are binned into an 125-bin histogram by considering that each of them can fall into 5 distinct bins, and the final histogram encodes in each bin a unique combination of the distinct values for each of the angles. One of these 125-bin histograms is produced for each input point.

There is also a version of PFH that includes color information: PFHRGB. This variant includes three more histograms, one for the ratio between each color channel of p and the same channel of q. These histograms are binned as the 3 angles of PFH and hence produce another 125 float values, giving the total size of 250 values for PFHRGB.

D. Fast Point Feature Histograms

The Fast Point Feature Histograms (FPFH) [10] are a simplification of the PFH descriptor above that reduce the computational complexity of the PPF algorithm from $O(nk^2)$ to $O(nk)$. The first step is to compute the histogram of the three angles between a point p and its k-nearest neighbors (not between all pairs of neighbors!) in the same way as in PPF. This produces the Simplified Point Feature Histogram (SPFH). Then, for each point p, the values of the SPFH of its k neighbors are weight by their distance $w_i = d$ to p to produce the FPFH at p: $FPFH(p) = SPFH(p) +$

$1/k \sum_{i=1}^{k} SPFH(i)/w_i$. The three angles are binned into 11-bin histograms that are concatenated into a single 33-bin FPFH descriptor. In [4], the authors found that using a different weighting scheme improves the recognition rates: $w_i = \sqrt{\exp \|d\|}$.

E. Point Pair Feature

Given two points p_1 and p_2 and their normals n_1 and n_2, the Point Pair Feature (PPF) [11] is given by: $PPF(p_1, p_2) = (\|d\|_2, \angle(n_1, d), \angle(n_2, d), \angle(n_1, n_2))$ where $\angle(a, b) \in [0, \pi]$ represents the angle between a and b and $d = p_2 - p_1$. The PPF is found for all pairs of points. The distances are sampled in d_{dist} steps and the angles in $d_{angle} = 2\pi / n_{angle}$ steps and the vectors with the same discrete representation are grouped.Consider M to be an object model and S to be the input scene. A global model descriptor is a mapping from the sampled space to the model space.

Consider that an arbitrary reference point $s_r \in S$ is chosen and assume it lies on a given object M. Then there is a point $m_r \in M$ that corresponds to s_r. If m_r and its normal are aligned with s_r and its normal, it is possible to align the model to the scene with a further rotation α around the aligned normal. The pair (m_r, α) is called the local coordinates of the model w.r.t. s_r.

A point pair $(m_r, m_i) \in M^2$ is aligned to a scene pair $(s_r, s_i) \in S^2$ that has the same feature vector. The transformation from the local model coordinates to scene coordinates is given by $s_i = T_{s \to g}^{-1} R_x(\alpha) T_{m \to g} m_i$.

To find the local coordinates that maximize the number of scene points that lie on the model, a voting scheme is used. This optimal local coordinate system allows the recovery of the global object pose. The poses obtained by the voting scheme are clustered according to how similar their rotations and translations are. The final pose is the one that corresponds to the average pose of the cluster with the largest sum of votes obtained by its members in the voting scheme.

F. Signature of Histograms of OrienTations

The SHOT descriptor [12] is based on obtaining a repeatable local reference frame using the eigenvalue decomposition around an input point. Given this reference frame, a spherical grid centered on the point divides the neighborhood so that in each grid bin a weighted histogram of normals is obtained. The descriptor concatenates all such histograms into the final signature. It uses 9 values to encode the reference frame and the authors propose the use of 11 shape bins and 32 divisions of the spherical grid, which gives an additional 352 values. The descriptor is normalized to sum 1. There is also a color version (SHOTCOLOR) proposed in [13] that adds color information (based on the CIELab color space) to the SHOT descriptor resulting in a 1344 value descriptor (plus 9 values to describe the local reference frame).

G. Unique Shape Context

The Unique Shape Context [14] was proposed as an upgrade of the 3DSC with the goal of avoiding the need to obtain as many versions of the descriptor as the number of azimuth bins. Consider a point p with a spherical neighborhood of radius R. A weighted covariance matrix M of the points in the neighborhood is computed as

$$M = \frac{1}{Z} \sum_{i:d_i \leq R} (R - d_i)(p_i - p)(p_i - p)^T$$

where p_i is a point in the spherical neighborhood, $d_i = \|p_i - p\|_2$ and $Z = \sum_{i:d_i \leq R} (R - d_i)$. The eigenvector decomposition of M is used to obtain the 3 unit vectors of the local reference frame. The sign of the eigenvectors with the biggest and smallest eigenvalues is changed so that it is coherent with the majority of the vectors they represents. The sign of the third eigenvector is obtained from the other two considering that they must form an orthonormal base. The eigenvector with the smallest eigenvalue gives the normal direction. Apart from this reference frame determination process, the USC descriptor is obtained like the 3DSC.

H. Viewpoint Feature Histogram

The Viewpoint Feature Histogram (VFH) [15] adds viewpoint variance to the above FPFH by using the viewpoint vector direction. It also produces only one descriptor for the input point cloud (it is a global descriptor). The process is the following: 1) find the input cloud centroid, c; 2) for each point p in the cloud, build the the local reference frame (u, v, w) using 2a) $u = n_c$; 2b) $v = (p - c) \times u$; 2c) $w = u \times v$; 3) Find the angles (α, ϕ, θ) as in the PFH, using this reference frame. Each of the three angles is binned into a 45-bin histogram. The angle $\beta = \arccos(n_p.c/\|c\|)$ that the central viewpoint direction translated to each normal makes with each point's normal is also encoded in a 128-bin histogram. The implemented version of this descriptor in PCL also uses a 45-bin histogram with the distances d between each point and the centroid. Thus the total length of the VFH descriptor is 308.

I. Clustered Viewpoint Feature Histogram

The Clustered Viewpoint Feature Histogram (CVFH) descriptor for a given point cloud dataset containing XYZ data and normals, was proposed in [5].

Stable regions are obtained by first removing the points with high curvature and then applying a smooth region growing algorithm.

The CVFH is obtained using the following steps: 1) determine the set S of stable regions; 2) for each $s_i \in S$, find the centroid (c) and its normal (n_c); 3) build a local reference frame (u_i, v_i, w_i) like in the VHF but using c and n_c instead of the centroid and respective normal for the whole input cloud; 4) find the histograms of the angles $(\alpha, \phi, \theta, \beta)$ as in VHF (the first 3 coded as 45-bin histograms and β coded as a 128-bin histogram); 5) find the Shape distribution Component (SDC) as $SDC = \frac{(c - p_i)^2}{\max\{(c - p_i)^2\}}$, $i = 1, \ldots, |S|$. The CVFH is given by the

concatenated histograms $(\alpha, \phi, \theta, SDC, \beta)$ which is a 308-bin histogram.

J. Ensemble of Shape Functions

This is a global shape descriptor proposed in [16] consisting of 10 concatenated 64-bin histograms resulting in a single 640 value histogram for a given input point cloud. It is based on three shape functions [17] describing distance (D2: distance between two randomly selected points), angle (A3: the angle enclosed by two lines created from 3 randomly selected points) and area (D3: area of the triangle formed by 3 randomly selected points) distributions. They also use an idea from [18] that is to classify each of the values into three classes based on where the connecting lines between points reside: on the object surface, off the surface and mixed (partly on and off). So the sub-histograms used to form the ESF are: 3 (on, off, mixed) for A3, 3 (on, off, mixed) for D3 , 3 (on, off, mixed) for D2 and a final one for the ratio of line distances D2 between off and on parts of each considered line. Before finding these distances the cloud is approximated by a voxel grid of side 64. The match is done using L_1-distance. The authors propose to optimize this descriptor by learning weights for the sub-histograms. In this paper we used the same weight for all sub-histograms.

K. Principal Curvatures Estimation

This descriptor calculates the directions (eigenvectors) and magnitudes (eigenvalues) of principal surface curvatures on each keypoint. It produces a 5 value descriptor on each point: 3 values for the principal curvature, which is the eigenvector with the largest eigenvalue, plus the largest and smallest eigenvalues. These are obtained using the cloud normals.

IV. EXPERIMENTS

A. Dataset

We used a subset of the large dataset of 3D point clouds from [19]. The original dataset contains 300 objects from 51 different categories captured on a turntable from 3 different camera poses. We used 48 objects representing 10 categories. Figure 2 shows one point cloud of one object from each of the 10 categories.

The training data contain clouds captured from two different camera views, and the test data contains clouds captured using a third different view. The training set has a total of 946 clouds while the test set contains 475 clouds. Since for each test cloud we do an exhaustive search through the complete training set to find the best match, this amounts to a total of 449.350 cloud comparisons for each of the evaluated descriptors and each of the keypoint extraction approaches.

B. Setup

To make a fair comparison between the descriptors, all steps in the pipeline are equal. All descriptors that use normals are evaluated with the normal estimation radius search equal to 1 cm. The Harris3D keypoint detector was used also with a radius search equal to 1.0 cm. The cloud sub-sampling was obtained using a voxelgrid with leaf size equal to 1 cm and 2 cm. Note that the fact that we sub-sample to 2 cm and have a normal estimation radius search equal to 1 cm is not incompatible since to find the normals at each of the sub-sampled points, the original (not sub-sampled) cloud is used. Some descriptors have several parameters: the values used were the ones set by default in PCL.

Fig. 2. One view of one object from each of the 10 categories used in the experiments. Left column, top to bottom: cap, calculator, binder, banana, apple. Right column, top to bottom: cell phone, camera, bowl, bell pepper, ball.

C. Results

Table II contains the experiments' results in terms of the best match, and figures 3 to 5 contain the recall $\times$ (1-precision) curves for the object recognition experiments.

The process of discriminating between different objects in the same category is much harder than the detection of the object's category since they objects from the same category can be very similar. This is clear from the difference in performance of all descriptors from the category to the object

TABLE II

CATEGORY AND OBJECT RECOGNITION ACCURACY USING THE HARRIS3D KEYPOINT EXTRACTOR AND CLOUD SUB-SAMPLING (WITH 1CM AND 2CM LEAF SIZE)

	Harris3D		Sub-sampl. 1cm		Sub-sampl. 2cm	
Descriptor	Category	Object	Category	Object	Category	Object
3DSC	74.05	32.91	87.34	50.63	78.06	40.51
CVFH	55.16	20.63	64.13	35.23	51.05	19.41
ESF	82.91	39.03	83.54	39.66	81.65	37.34
FPFH	81.89	44.63	87.55	49.58	86.08	47.26
PFH	86.95	48.42	89.87	56.75	89.24	55.49
PFHRGB	**93.89**	**77.89**	**94.09**	**79.32**	**94.73**	**79.75**
PCE	37.89	9.26	50.84	17.09	47.26	16.46
PPF	37.47	8.63	56.33	18.14	52.32	17.09
RIFT	34.11	9.68	75.11	35.44	60.34	24.05
SHOT	81.26	42.53	91.77	55.49	88.19	46.84
SHOTCOLOR	88.40	69.20	92.62	75.53	90.72	73.42
USC	73.00	35.44	86.50	51.05	82.07	45.78
VFH	28.84	6.11	52.11	23.63	21.73	5.27
Average	65.83	34.18	77.83	45.20	71.03	39.13

TABLE III

SIZE OF THE FEATURE DATABASE IN MEGABYTES AND TIME TO PROCESS THE TEST SET IN SECONDS USING THE HARRIS3D KEYPOINT EXTRACTOR AND CLOUD SUB-SAMPLING (WITH 1CM AND 2CM LEAF SIZE)

	Harris3D		Sub-sampl. 1cm		Sub-sampl. 2cm	
Descriptor	Size	Time	Size	Time	Size	Time
3DSC	143	475	1011	1206	281	504
CVFH	1.1	15	2.1	13	1.4	13
ESF	5.7	25	5.8	15	5.7	15
FPFH	12	241	69	240	21	228
PFH	21	1054	131	6049	40	1668
PFHRGB	40	1883	249	10753	76	2992
PCE	1.8	77	11	36	3.3	13
PPF	158	92	5400	1550	461	153
RIFT	7.8	15	68	19	17	14
SHOT	50	112	310	175	94	76
SHOTCOLOR	159	121	987	676	297	178
USC	142	184	1014	1195	285	381
VFH	0.8	15	1.8	14	1	13
Average	57.1	331.5	712.3	1687.8	121.8	480.6

recognition experiments.

Two sizes for the sub-sampling process were evaluated. Both give more keypoints than the ones obtained from the Harrys 3D keypoint detector. The average number of keypoints extracted in the training set from these three approaches was: 40.73 (Harris), 249.56 (sub-sample 1cm) and 75.18 (sub-sample 2cm).

We can see by the average value for each column in table II that the number of extracted points has a big influence on the recognition performance: the larger the number of points extracted, the better the algorithms perform, on average. So, the results using sub-sample 1cm are the best and the ones using Harrys 3D keypoint detector the worst, on average.

It is interesting to see that even though the data obtained from the sub-sampling with 2cm has around twice the number of points that the Harrys 3D keypoint detector produces, results using the later sometimes outperform the results obtained with the former. This happens with the CVFH, ESF and VHF in both the category and object recognition tasks. This is possibly the result of the Harrys 3D keypoints being more "meaningfull" whereas the sub-sampled keypoints are "blind" to the type of region they are in: even in regions that show no relevant information there will be sub-sampled keypoints with exactly the same density as the keypoints obtained on more interesting object regions, such as those with edges or other rapidly changing features (such as curvature or color). This makes us think that if the same number of keypoints is extracted using the Harrys 3D detector and a sub-sampling approach, the former will yield better results than the latter.

The best results were obtained in both the keypoint and sub-sampled experiments with the PFHRGB. It is interesting to compare it to the PFH: the differences in performance can only be attributed to the use of color in the former. The same is true for the SHOTCOLOR versus the SHOT descriptor. The importance of color can also be seen in the fact that the two descriptors that use it present always the two best results in terms of both category and object recognition. The curves in figures 3 to 5 also clearly show the superiority of these two descriptors versus the remaining.

Another interesting result is the one obtained by the FPFH when compared to the PFH: as the proposers of this descriptor suggested, it has a performance slightly worst than the performance of the PFH, but it is faster to extract and uses about half the space (check table III).

The USC was proposed as an upgrade to the 3DSC and our results confirm that in fact it improves the 3DSC results in all three object recognition tasks; but in the category tasks, the 3DSC beats the USC, by a small amount, when using the Harris and the 1cm sub-sampling.

We must stress that the obtained recognition results are dependent of the used set distance. As such, the relative performance of the presented descriptors can change if other set distances are used.

Table III presents training database size and the time using a i7-3930K@3.2GHz CPU on Fedora 17. In terms of time, the best descriptor is also the most costly by a very large difference. The SHOTCOLOR descriptor seams to have a very good balance between recognition results and time complexity.

The descriptor's requirements varies a lot: the difference in terms of time can be as much as 800 times, and in terms of space as high as 3.000 times. This tells us that if the application needs real-time performance there are some descriptors that cannot be considered; same thing applies in terms of space, for instance, when using embedded devices with limited resources.

V. CONCLUSIONS

In this paper we focused on the available descriptors on the PCL library, explaining how they work, and made a comparative evaluation on public available data. The main conclusions are: 1) increasing the number of keypoints improves recognition results at the expense of size and time;

2) since there are big differences in terms of recognition performance, size and time requirements, the descriptor should be matched to the desired task; 3) a descriptor that uses color information should be used instead of a similar one that uses only shape information; 4) the SHOTCOLOR descriptor presents a good balance between recognition performance and time complexity.

REFERENCES

[1] R. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[2] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Vision Conference*, Manchester, 1988, pp. 147–152.

[3] A. E. Johnson and M. Hebert, "Surface matching for object recognition in complex three-dimensional scenes," *Image Vision Comput.*, vol. 16, no. 9-10, pp. 635–651, 1998.

[4] R. Rusu, A. Holzbach, and M. Beetz, "Detecting and segmenting objects for mobile manipulation," in *S3DV Workshop of the 12th International Conference on Computer Vision (ICCV)*, 2009.

[5] A. Aldoma, N. Blodow, D. Gossow, S. Gedikli, R. Rusu, M. Vincze, and G. Bradski, "Cad-model recognition and 6 dof pose estimation," in *ICCV 2011, 3D Representation and Recognition (3dRR11) workshop*, 2011.

[6] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik, "Recognizing objects in range data using regional point descriptors," in *8th European Conference on Computer Vision*, 2004, pp. 224–237.

[7] S. Lazebnik, C. Schmid, and J. Ponce, "A sparse texture representation using local affine regions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 1265–1278, 2005.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, November 2004. [Online]. Available: http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94

[9] R. Rusu, N. Blodow, Z. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," in *International Conference on Intelligent Robots and Systems (IROS)*, Nice, France, September 22-26 2008.

[10] R. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *International Conference on Robotics and Automation (ICRA)*. Kobe, Japan: IEEE, May 12-17 2009.

[11] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3d object recognition," in *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010*, San Francisco, CA, USA, 2010, pp. 998–1005.

[12] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *Proceedings of the 11th European conference on computer vision conference on Computer vision: Part III*. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 356–369.

[13] ——, "A combined texture-shape descriptor for enhanced 3D feature matching," in *IEEE International Conference on Image Processing*, September 2011.

[14] ——, "Unique shape context for 3d data description," in *Proceedings of the ACM workshop on 3D object retrieval*, ser. 3DOR '10. New York, NY, USA: ACM, 2010, pp. 57–62. [Online]. Available: http://doi.acm.org/10.1145/1877808.1877821

[15] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram," in *Proceedings of the 23rd IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, October 18-22 2010.

[16] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3D object classification," in *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*, December 2011, pp. 2987 –2992.

[17] R. Osada, T. A. Funkhouser, B. Chazelle, and D. P. Dobkin, "Matching 3D models with shape distributions," in *International Conference on Shape Modeling and Applications*, Genoa, Italy, 2001, pp. 154–166.

[18] C. Y. Ip, D. Lapadat, L. Sieger, and W. C. Regli, "Using shape distributions to compare solid models," in *Symposium on Solid Modeling and Applications*, 2002, pp. 273–280.

[19] K. Lai, L. Bo, X. Ren, and D. Fox, "A Large-Scale hierarchical Multi-View RGB-D object dataset," in *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2011.

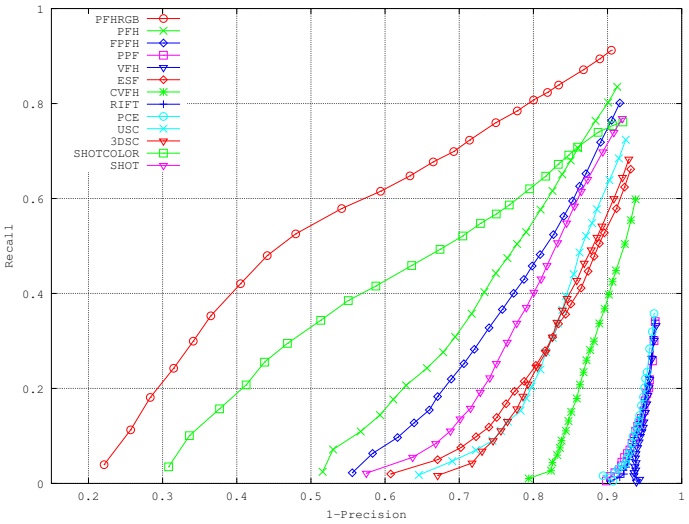

Fig. 3. Recall × (1-Precision) curves for the object recognition experiments using the Harris3D keypoints (best viewed in color).

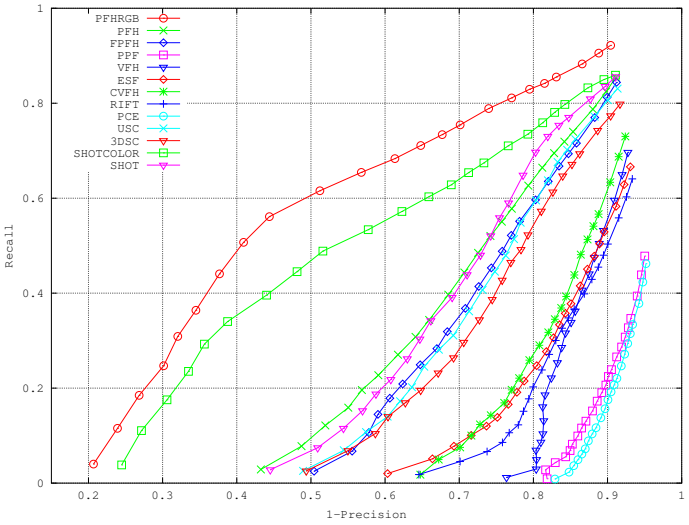

Fig. 4. Recall × (1-Precision) curves for the object recognition experiments using the sub-sampled (1 cm) keypoints (best viewed in color).

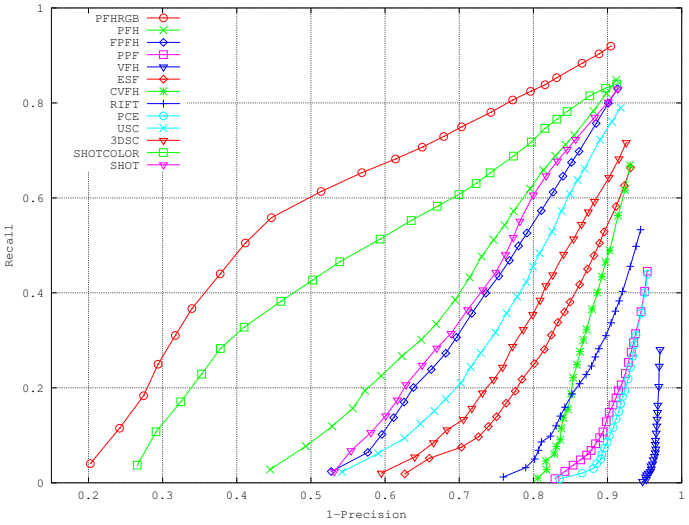

Fig. 5. Recall × (1-Precision) curves for the object recognition experiments using the sub-sampled (2 cm) keypoints (best viewed in color).