CAD-Model Recognition and 6DOF Pose Estimation Using 3D Cues

Aitor Aldoma and Markus Vincze
ACIN - Technische Universitat Wien
aldoma,vincze@acin.tuwien.ac.at

Nico Blodow, David Gossow, Suat Gedikli, Radu Bogdan Rusu and Gary Bradski
Willow Garage

Abstract

This paper focuses on developing a fast and accurate 3D feature for use in object recognition and pose estimation for rigid objects. More specifically, given a set of CAD models of different objects representing our knoweledge of the world - obtained using high-precission scanners that deliver accurate and noiseless data - our goal is to identify and estimate their pose in a real scene obtained by a depth sensor like the Microsoft Kinect. Borrowing ideas from the Viewpoint Feature Histogram (VFH) due to its computational efficiency and recognition performance, we describe the Clustered Viewpoint Feature Histogram (CVFH) and the cameras roll histogram together with our recognition framework to show that it can be effectively used to recognize objects and 6DOF pose in real environments dealing with partial occlusion, noise and different sensors atributes for training and recognition data. We show that CVFH outperforms VFH and present recognition results using the Microsoft Kinect Sensor on an object set of 44 objects.

1. Introduction and related work

Object recognition and pose estimation is a well studied problem in computer vision due to its endless applications in scene understanding, robotics, virtual reality, *etc*. Several feature descriptors for object recognition have been presented in the literature, both in 2D (e.g. [6]) and 3D (e.g. [12]). However, they still can not manage to resolve the full object recognition problem, especially when faced with hard problems such as textureless objects noise or missing parts of the objects. For both 2D and 3D, there are mainly two different approaches to the object recognition problem: local (e.g. [1],[2],[4]), or global descriptors (e.g. [5],[8]).

The latter and most relevant in the scope of the paper, describe the geometry, appearance or both of a whole partial view of an object and are more robust to noise than local features, specially in the 3D domain but they require the notion of object before recognition which is normally given by a prior segmentation procedure. Because of its global nature they have problems dealing with missing parts which are caused by partial occlusions, sensor limitations or segmentation artifacts (see Figure 1).

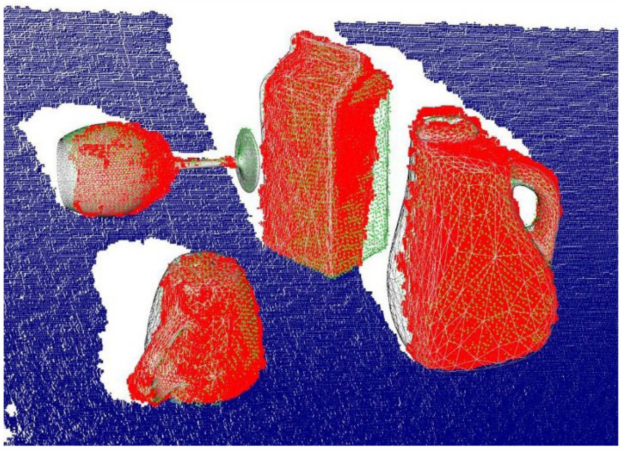
Figure 1. The figure shows how CVFH can deal with limited amounts of occlusion. The support plane is shown in blue, the segmented object candidates from the current scene in red, the recognized views from the database in green and the corresponding models overlapped as grey meshes.

These artifacts increase the complexity of the problem, mostly in our specific scenario where we want to develop a feature that can be trained on 3D CAD models and yet perform recognition on real data. Almost none of the descriptors presented in the literature (excepting [13]) have tackled the problem of training on synthetic data and matching on real data. Creating training databases for a reasonable number of objects using real devices can be a cumbersome task, even very difficult if one would like to have all different viewpoints and poses of an object. On the other hand, there are publicly available databases of CAD models and accurate 3D meshes for thousands of objects found in our daily life (e.g., Google Warehouse). Given a 3D CAD model and a rendering system, it is straightforward to place a virtual camera around the object and obtain all desired viewpoints without the need of calibrated systems and a time-consuming capturing process. We believe this is a crucial factor for cost and ease of scaling the set of objects the robot can learn and manipulate.

With the advent of the Kinect, depth information at ranges of 0.8-3.5 meters can be obtained at framerate and at a moderate price. This cost breakthrough is an enabler for vision systems and robotics and new low cost depth sensors such as the WAVI Xtion [9] are following rapidly. Now that depth information is cheap and easy to get, there is a need to develop efficient 3D features that will work effectively with this data in order to support robot object recognition plus 6DOF pose for manipulation.

We decided to build on the VFH feature which is efficient to compute, and already showed high discriminability in previous work [8]. As described below, VFH has shortcomings to perform recognition on real data when trained on synthetic data.

The rest of the paper is organized as follows: In section 2 the Viewpoint Feature Histogram is reviewed and used to motivate the Clustered Viewpoint Feature Histogram (CVFH), presented in section 3, that meets our goal of allowing for training on 3D CAD models and yet performing well on real world data. In section 4, the Camera's Roll Histogram is presented as an efficient way to deal with the invariance to rotations around the camera axis that appear in 3D global descriptors based on partial views. In section 5, we present the recognition framework allowing for training and recognition which includes the histogram metric used for nearest neighbor searches together with the post-processing applied after CVFH recognition to refine the results. In section 6, we compare CVFH against VFH and show that CVFH outperforms it. Finally, we conclude in section 7 and present our future work lines.

2. The Viewpoint Feature Histogram

The VFH descriptor is a compound histogram representing four different angular distributions of surface normals. Let p_c and n_c be the centroids of all surface points and their normals of a given object partial view in the camera coordinate system (with $||\mathsf{n}_\mathsf{c}|| = 1$). Then $(\mathsf{u}_i, \mathsf{v}_i, \mathsf{w}_i)$ defines a Darboux coordinate frame for each point p_i (see [10]):

$$\begin{aligned} \mathsf{u}_i &= \mathsf{n}_\mathsf{c} \\ \mathsf{v}_i &= \frac{\mathsf{p}_\mathsf{i} - \mathsf{p}_\mathsf{c}}{||\mathsf{p}_\mathsf{i} - \mathsf{p}_\mathsf{c}||} \times \mathsf{u}_i \\ \mathsf{w}_i &= \mathsf{u}_i \times \mathsf{v}_i \end{aligned} \tag{1}$$

The normal angular deviations $\cos(\alpha_i)$, $\cos(\beta_i)$, $\cos(\phi_i)$ and θ_i for each point p_i and its normal n_i are given by:

$$\begin{aligned} \cos(\alpha_i) &= \mathsf{v}_i \cdot \mathsf{n}_\mathsf{i} \\ \cos(\beta_i) &= \mathsf{n}_\mathsf{i} \cdot \frac{\mathsf{p}_\mathsf{c}}{||\mathsf{p}_\mathsf{c}||} \\ \cos(\phi_i) &= \mathsf{u}_i \cdot \frac{\mathsf{p}_\mathsf{i} - \mathsf{p}_\mathsf{c}}{||\mathsf{p}_\mathsf{i} - \mathsf{p}_\mathsf{c}||} \\ \theta_i &= \operatorname{atan2}(\mathsf{w}_i \cdot \mathsf{n}_\mathsf{i}, \mathsf{u}_i \cdot \mathsf{n}_\mathsf{i}) \end{aligned} \tag{2}$$

Note that $\cos(\alpha_i)$, $\cos(\phi_i)$ and θ_i are invariant to viewpoint changes, given that the set of visible points does not change. For $\cos(\alpha_i)$, $\cos(\phi_i)$ and θ_i histograms with 45 bins each are computed and a histogram of 128 bins for $\cos(\beta_i)$, thus the VFH descriptor has 263 dimensions.

Though VFH showed promising results in [10], it has a few shortcomings:

- it is invariant to the size of the object as the compound histogram is normalized by the total number of points in the partial view;
- it is invariant to rotations around the camera's view direction, so it does not allow full pose estimation;
- using the centroid and average normals (p_c and n_c) to build the Darboux coordinate system, makes VFH sensitive to missing parts of the object caused by partial occlusions, segmentation or sensor artifacts.

3. The Clustered Viewpoint Feature Histogram

As outlined in section 2, the major flaws to VFH are its sensitivity to noise and occlusions (e.g. missing parts of the object) and the fact that it is invariant to rotations about the camera axis. By analyzing the data obtained from the Kinect, we noticed that surfaces that are at a steep angle relative to the sensor as well as parts that are close to object borders contain more noise or even miss a few depth estimates (see Figure 2).

These effects can result in unstable estimations of the object points and normals centroid (p_c and n_c from Eq. 1), thus affecting the resulting VFH and making it unsuitable to match against the corresponding synthetic view that will not present these artifacts.

The main idea behind CVFH is to take advantage from the object parts that can be robustly estimated by the depth sensor and use them to build the Darboux coordinate system while still using the whole partial view to compute the descriptor.

Formally, we propose to describe a partial view of an object, represented by a set of points $\mathcal{P}$, as a set $\mathcal{H}$ of Clustered Viewpoint Feature Histograms. The cardinality of $\mathcal{H}$ is the same as the cardinality of $\mathcal{S}$, where $\mathcal{S}$ is the set of stable regions found on $\mathcal{P}$ using the procedure defined in the upcoming section 3.1.

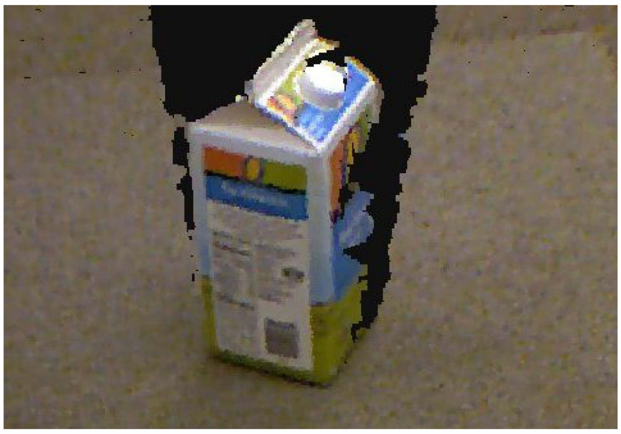
Figure 2. Example of an incomplete surface due to limitations of the sensor.

Taking $\mathsf{s}_i \in \mathcal{S}$ with $\mathsf{s}_i \subseteq \mathcal{P}$, we can define a Darboux coordinate system $\mathcal{D} = (\mathsf{u}_i, \mathsf{v}_i, \mathsf{w}_i)$ like in Eq. 1 but in this case p_c and n_c represent the euclidean centroid and normal centroid of s_i and not of the whole partial view $\mathcal{P}$. Given $\mathcal{D}$ and using Eq. 2, the normal angular deviations for all points in $\mathcal{P}$ can be computed.

Let then $(\alpha, \phi, \theta, \beta)$ represent the normal angular deviations already bined in (45,45,45,128) bins, the CVFH histogram $\mathsf{h}_i \in \mathcal{H}$ is defined as the following concatenation:

$$(\alpha, \phi, \theta, \mathcal{SDC}, \beta) \tag{3}$$

where $\mathcal{SDC}$ represents the Shape Distribution Component of CVFH computed as follows:

$$\mathcal{SDC} = \frac{(\mathsf{p}_c - \mathsf{p}_i)^2}{\max((\mathsf{p}_c - \mathsf{p}_i)^2)} \tag{4}$$

The number of bins used for this component is again 45 thus making a total size of 308 for CVFH. This component allows to differentiate surfaces that have very similar normal distributions and sizes but their points are distributed differently. For instance we could differentiate an elongated planar surface from a more compact planar surface.

To avoid scale invariance, each bin in CVFH count the absolute number of points falling in that bin. To reduce ambiguities, we first construct a voxel grid over our point cloud data with a fixed voxel size, and reduce the cloud to the set of voxel centroids. Because the actual size of the object is given by the 3D sensor, the amount of points for a given view will be the same no matter what the distance to the camera is. Avoiding the normalization step allows us to distinguish between objects of different size but identical shape. It also makes the descriptor more robust to missing parts of the object, as this will only influence local parts of the descriptor (compare Figure 3). Normalizing the histogram by the total number of points would increase the bins height under the presence of occlusion.

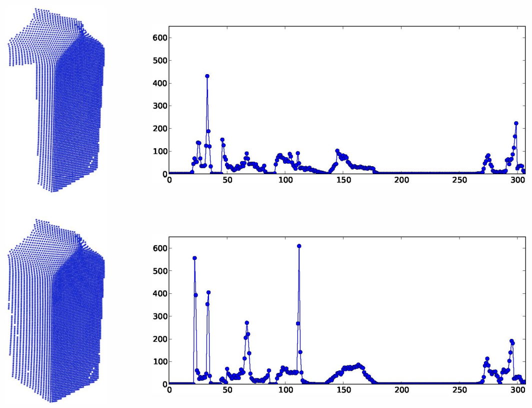

Figure 3. The CVFH histograms become additive when the centroids are consistent. *top:* Missing part on the view and the correspondent CVFH signature. *bottom:* Whole view and and the correspondent CVFH signature.

The advantages of CVFH are two-fold: (i) the coordinate system is more likely to resemble the one obtained from the synthetic view making the descriptor more stable and (ii) because the set of CVFHs represent a multivariate description of the partial view, we can better handle occlusions as long as any of the stable region is visible. Please note that the CVFH histograms in H are independent from each other and not complementary as they describe the same geometry but encode them differently. To understand how CVFH is used for recognition, we refer the reader to the next section (Recognition Framework).

3.1. Stable regions clustering

To overcome the instability caused by missing object parts and local noise artifacts, we first identify stable regions in partial view obtained by the depth sensor. To do so, we apply a smooth region growing algorithm on the points obtained from a partial view of an object after removing points with high curvature (caused by noise, object edges or non-planar patches).

Each new cluster is initialized with a random point. A point p_i with normal n_i is added to a cluster C_k if the cluster contains a point p_j with normal n_j in the direct neigbourhood of p_i with a similar normal, i.e. the following constraint is fulfilled:

$$\exists p_j \in C_k : ||p_i - p_j|| < t_d \wedge n_i \cdot n_j > t_n \tag{5}$$

For our experiments, t_d is set to three times the voxel grid size and t_n to $\cos(10°)$. For each stable region, a CVFH descriptor is computed as outlined in the previous section. The number of stable regions for a specific partial view defines the cardinality of the descriptor set $\mathcal{H}$.

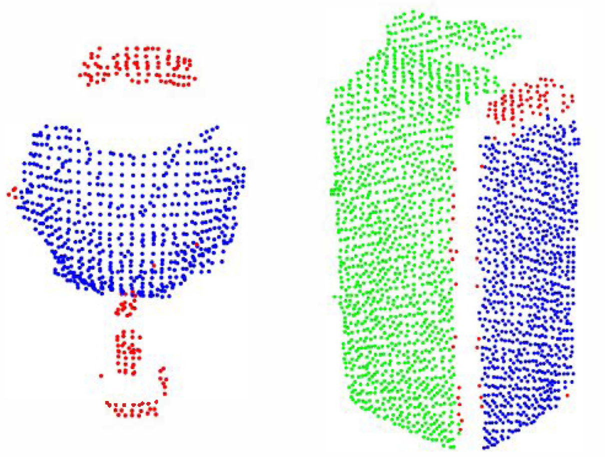

Figure 4. Free shape smooth clustering. *Left:* a wine glass, and *right:* a milk carton. Smooth surfaces are clustered together. Points in red do not belong to any cluster and points with high curvature are not shown, e.g. at the edges of the milk carton.

Only regions with more than 50 points in total are considered to be stable and taken into account. In the case that no regions are found that fulfill these conditions, the CVFH centroids are computed using all points in the partial view.

Intuitively, we are trying to define a stable location to base the computation of the CVFH descriptor even when parts of the objects are missing. For instance, the base and stem of the wine glass in Figure 4 is partly missing, which usually happens due to oversegmentation of its support plane in an earlier processing step. This will affect the descriptor centroids if the complete partial view is used, but the stable region shown in blue remains unchanged. In the case of the milk carton where 2 stable regions are found, the centroid for one of the dominant surfaces stays stable when the other one is occluded and thus the stable CVFH will allow for a positive recognition (see Figure 1 where part of the milk cartoon is occluded).

4. Camera roll histogram and 6DOF pose

Most descriptors based on views of an object like VFH, CVFH, CAP-SIFT [3] are unable to deliver a complete 6-DOF pose. Due to the this invariance of CVFH with respect to rotations along the view direction of the camera (roll), the object and viewpoint recognition is determined up to an unknown rotation. To determine the correct orientation of the object, we introduce a new descriptor that is not invariant to the roll angle. To avoid a higher dimensionality in the overall descriptor by extending it, which would decrease the performance of the object/viewpoint recognition noticeably, we use a final optimization step to find the correct roll angle. Since the computation of the roll angle is only done for the best N candidates from the CVFH matching step and furthermore is efficient to calculate, the overall performance is not affected drastically.

For each CVFH descriptor in $\mathcal{H}$, an additional histogram is computed - *the camera's roll histogram*. We project the normals at each point onto a plane that is orthogonal to the vector given by the camera center and the centroid of the stable region used to compute CVFH. For the projection, we compute a rotation-axis v and a rotation angle θ using Eq. 6 that transforms the CVFH centroid $\mathsf{p_c}$ to coincide with the camera's z-axis. Since we use an orthographic projection, the projected normals are given by the first two components of the transformed normals n_i.

$$\begin{aligned} v &= \frac{p_c \times z}{||p_c||} \\ \theta &= -\arcsin\left(||v||\right) \end{aligned} \tag{6}$$

The roll histogram is then computed by taking the angle of the projected normal relative to the up-view vector of the camera on the plane. The histogram contains 90 bins giving an angular resolution of 4 degrees. The number of bins for the camera-roll-histogram is selected from our empirical evaluations to provide a reasonable trade off between efficiency and accuracy. Due to noise in the input data, we weight the projected normals by their magnitudes. This removes most of the equally distributed noise in the histogram, resulting from unstable projections of normals that are almost parallel to the roll axis of the camera.

Figure 5 shows two histograms of the same object. The upper one is from the object in upright orientation, whereas the bottom histogram is computed from the object rotated around the roll axis by 44°.

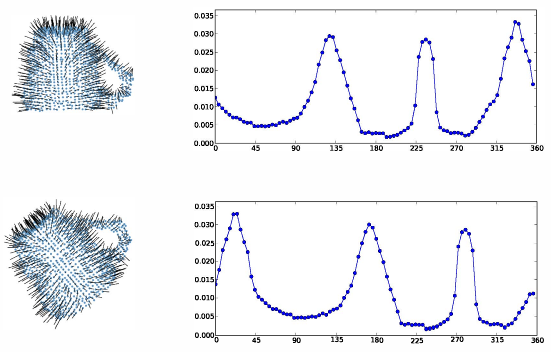

Figure 5. The camera roll histograms of the same object in different orientations.

In order to estimate the object's rotation around the roll axis, we need to find an orientation where the two roll histograms match best according to a metric. This can be considered a correlation maximization problem. Therefore, we apply a Discrete Fourier Transform for both histograms, and multiply the complex coefficients of the database view with the complex conjugate coefficients, and perform the inverse transform to compute the cross power spectrum R. The peaks of this spectrum appear at rotation angles that align the two roll histograms well.

There are cases where the power spectrum of two roll histograms can have multiple high peaks due to different kinds of symmetries. Also, partial occlusions or sensor noise might deteriorate the roll histograms, so it is generally not sufficient to rely solely on the maximal peak in R.

In order to select a set of orientations that can be pruned in a subsequent test, we select a minimum threshold t_p for peaks, and add peaks with higher magnitude to the set. We start with the highest peak, adding peaks if their corresponding rotation angles do not fall within a certain distance band t_d of any of the previously added peaks. This ensures that the set of orientations does not contain multiple entries for very similar alignments, but captures local maxima that are distributed over the whole set of rotations, if they indicate a good alignment.

In our experiments, we set $t_d = 12°$ and chose a relatively high value for t_p in order to keep the size of the rotation set small. We found a value of $t_p = 0.9 * \max(R)$ to yield a low number of peaks - typically up to 10 peaks - while still capturing corner cases.

5. Recognition framework

In this section, we concentrate on the recognition framework which consists of two different parts: an offline training stage where the CVFH descriptors are computed for the models in our training set, and an online recognition stage, in which the real scene is processed. The recognition stage includes segmentation, recognition and pose estimation using CVFH and final refinement of the recognition results. Please note, that in this case, segmentation refers to finding possible objects candidates in the scene and not to the stable regions clustering step presented before.

5.1. Training stage

Our training data is generated from a set of CAD models. Because CVFH works on views from object, our first step is to take each of the CAD models and generate a set of distinguishable views. We place a virtual camera on the vertices of a tesselated sphere looking at the CAD model of the object and render the object seen from that viewpoint into a depth buffer from which we can efficiently extract a partial pointcloud.

For each view, the CVFH descriptor and roll histogram is computed. Views that are not distinguishable, like symmetric objects as bottles or bowls, are not considered, reducing the initial number of 80 view to about 12 views per object.

To decide which views can be removed, we align two different views of the same object using the camera's roll histogram and compute the overlapping between the aligned point clouds by searching for each point in one of the views the nearest neighbor in the other view. A point is considered not to overlap if the nearest neighbor is not within a range of twice the voxel grid size. If more than 2.5% do not overlap the view is considered to be different and the next view is checked.

5.2. Recognition stage

The recognition stage runs on a raw pointcloud from a depth sensor, which in our case is the Kinect. We proceed first with a segmentation of the scene using dominant plane extraction and Euclidean segmentation on the remaining points [11]. The segmented groups of points represent the objects to be recognized. Independently for each object in the scene:

1. Compute a set of CVFH descriptors (H) and camera's roll histograms. Please note, that each CVFH descriptors is paired with a camera roll histogram.
2. For each CVFH in H, a nearest neighbor (NN) search is performed to find the N closest CVFH descriptors in the training set, giving a set of views from the trained objects.
3. As we have performed as many NN-searches as elements in H, the best N candidates according to the metric given in Eq. (7) are selected.
4. For the resulting N view candidates the roll angle is determined using the roll histogram matching and 6DOF pose estimation (as detailed in section 4).
5. After aligning the views using the pose and roll information gathered so far, an additional ICP [14] step is used to refine the alignment.
6. Finally the N best view candidates are sorted using the number of inliers from the last iteration of ICP using a distance threshold of twice the voxel grid size.

Because of its efficiency, we use the FLANN library [7] to perform the nearest neighbor search. FLANN includes different search and indexing methods such as linear search, randomized kd-trees or hierarchical k-means indexing. Moreover, it provides different distance and histogram comparison metrics for high dimensional spaces, including e.g. L1, L2, Histogram Intersection, and ChiSquare.

We have performed different empirical experiments to determine which is the best metric for our needs. The major problem with metrics like L1 and L2 are its sensitivity to outliers. Dealing with partial occlusions implies that the histograms will have outliers due to missing parts of the objects even if the rest of the histogram is shaped correctly. Therefore, we use the following metric:

$$d(A,B) = 1 - \frac{1 + \sum_{i=1}^{308} \min(A_i, B_i)}{1 + \sum_{i=1}^{308} \max(A_i, B_i)}, \quad (7)$$

where A and B represent two CVFH descriptors. This metric is not element-wise addivite, making it unsuitable for kd-tree search but suitable for hierarchical k-means indexing. At the moment, we are using linear search to retrieve the nearest neighbor since our database contains only 1704 CVFH descriptors for the 44 objects in our training set. The computation time for finding the nearest neighbor is below $2ms$ in our experiments, and using other search methods such as hierarchical indexing requires an addiotional overhead to construct the appropriate search structure, which is not necessary for linear search.

6. Results

For the evaluation of CVFH, we perform a different set of experiments and compare the results to VFH. First, we evaluate the performance on our training set for noise. We also evaluate the performance of both descriptors in matching single objects in real scenes obtained with the Kinect sensor. Finally, we show some scenes with the aligned models overlapped as a qualitative evaluation, see Figure 7.

The criteria we use to evaluate performance in synthetic data are multiple:

- Correct view and correct object id, respectively, in the first result.
- Correct view and correct object id, respectively, in the first N results.
- To test the performance of the camera's roll histogram, all views from the training set are randomly rotated along the virtual camera's roll axis. Because of discretization errors, we assume the result to be correct if the computed angle is off by 4° or less from the applied rotation.

6.1. Noise

Each view in the training set is noisified by applying a Gaussian kernel to each point. We use different standard deviations to test robustness to noise, ranging from 0.5mm to 2mm for each point in the view. After a view is noisified, we compute the CVFH and VFH and perform a search for the nearest neighbors in our descriptors database obtained from non-noisified views and compute the metrics listed above. Table. 1 and Table. 2 show respectively the results for VFH and for CVFH.

Table. 1 and Table. 2 show that with this kind of uniform noise and without missing parts VFH performs better than

	Noise levels (Stdev in mm)			
	0.5mm	1mm	1.5mm	2mm
View (1st)	99.32%	97.25%	92.76%	86.39%
View (N-1st)	100%	100%	99.93%	99.60%
Roll	98.32%	96.51%	93.89%	90.84%
Id (1st)	99.53%	97.98%	94.63%	89.87%
Id (N-1st)	100%	100%	100%	99.79%

Table 1. Recognition rates and roll angles correctness with different amount of noise applied on the training data using VFH.

	Noise levels (Stdev in mm)			
	0.5mm	1mm	1.5mm	2mm
View (1st)	93.89%	94.63%	86.92%	41.51%
View (N-1st)	97.38%	97.58%	93.62%	59.96%
Roll	97.10%	97.52%	94.48%	79.53%
Id (1st)	97.45%	97.38%	94.29%	58.62%
Id (N-1st)	99.53%	99.73%	99.46%	77.33%

Table 2. Recognition rates and roll angles correctness with different amount of noise applied on the training data using CVFH.

CVFH. Because in CVFH, the amount of points used for the computation of the centroid and the average of the normals which are used to build the Darboux coordinate system is usually smaller than in VFH, CVFH becomes more sensitive to this noise applied uniformly over the whole partial view. Another reason is that when the amount of noise increases, the estimation of stable regions becomes very unstable thus making the CVFH descriptor also unstable.

It is interesting to note that the roll orientation performs extremely well (over 90% with 1.5mm noise) when CVFH or VFH return the correct view.

6.2. Recognition and pose evaluation on real scenes

We have performed recognition experiments on 18 of our 44 objects in the database to estimate the recognition rate of CVFH, VFH and CVFH + post-processing using Kinect data. Here, we refer to CVFH + post-processing as the steps 5) and 6) outlined in section 5.2. Because ground truth data for pose is not easily obtained, we decided to evaluate the recognition results manually.

To do so we have taken each of the 18 objects independently and placed them in the field of view of the sensor. The cluster of points representing the view of the object is extracted using Euclidean segmentation and recognized using CVFH, VFH and CVFH + post-processing to refine the recognition results. The recognition of the three pipelines are displayed together with the matching view in the database and the CAD model overlayed. All three recognitions include the computation of the roll orientation for a full 6DOF pose. We visually inspected the results and annotated independently for each 3 results set at which position the correct object and pose is found. For each recognition, 14 nearest neighbors were retrieved. Each object was recognized 10 times in different stable poses.

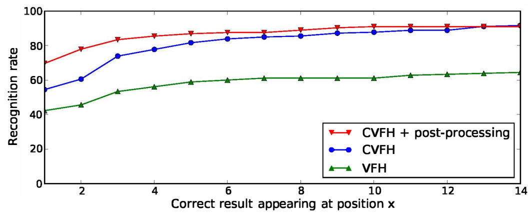

Figure 6. Recognition rate for CVFH, VFH and CVFH including the recognition framework. We show how often the correct solution appeared within the first x results.

Figure 6 shows the result of the experiment, where each point represents how many times we identified the right object with in the first x results. It can be seen that CVFH outperforms VFH both in recognition rate for the first result and for the accumulated recognition rate over the first 14 results. This is to show that although the recognition rate just by looking at the first result of $CVFH$ is below 60% we obtain the correct solution in the top-10 results in 90% of the cases. If we take into account the top-10 results of CVFH which include the right solution in 90% of the cases and sort these with post-processing, we increase the recognition rate in the first result to 70% of the cases. Ideally, we would like the recognition rate after post-processing to be 90% meaning that the post-processing can always identify the right solution if available in the candidates given by CVFH. In this case, for a desired recognition rate of 90%, the number of candidates is reduced using CVFH from 1409 (number of views) to 10.

7. Conclusions and future work

We have presented the Clustered Viewpoint Feature Histogram (CVFH) and shown that it can be robustly used to recognize objects and detect their poses in real scenes even when the training data source has different properties. In the scope of the paper, 44 objects were trained using CAD models and recognized in real scenes using the Kinect sensor.

Being able to determine a stable normal and a stable centroid on the objects allows us to deal with partial occlusions and handle properly the different properties of the training and recognition sensors. In our experiments we have shown that CVFH returns in 90% of the cases the correct view in the first 10 results reducing the number of candidates that need to be processed from approx. 1409 (number of views) to 10 in less than 2ms.

We have also presented the camera's roll histogram that can efficiently compute the rotation about the roll axis of the camera to which CVFH is invariant.

Future work includes dealing more robustly with higher degrees of clutter and occlusion, larger object databases and taking advantage of the semi-global nature of CVFH be able to solve undersegmentation issues.

References

[1] M. B. Ajmal S. Mian and R. Owens. Three-dimensional model-based object recognition and segmentation in cluttered scenes. *Ieee Transactions On Pattern Analysis And Machine Intelligence, Vol. 28, No. 10, October*, 2006. 1

[2] K. K. B. Steder, R. B. Rusu and W. Burgard. Narf: 3d range image features for object recognition. *In Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010. 1

[3] C. Goldfeder, M. Ciocarlie, J. Peretzman, H. Dang, and P. K. Allen. Data-driven grasping with partial sensor data. 4

[4] A. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, May 1999. 1

[5] X. R. Kevin Lai, Liefeng Bo and D. Fox. A large-scale hierarchical multi-view rgb-d object. *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. 1

[6] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004. 1

[7] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pages 331–340. INSTICC Press, 2009. 5

[8] M. Muja, R. B. Rusu, G. Bradski, and D. G. Lowe. Rein - a fast, robust, scalable recognition infrastructure. In *ICRA 2011*, Shanghai, China, May 2011. 1, 2

[9] PrimeSense. Primesense teams up with asus to bring intuitive pc entertainment to the living room with wavi xtion. In *Business Wire*, 2011. 2

[10] 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, 10/2010 2010. 2

[11] R. B. Rusu, A. Holzbach, M. Beetz, and G. Bradski. Detecting and segmenting objects for mobile manipulation. In *ICCV S3DV workshop*, 2009. 5

[12] R. B. Rusu, Z. C. Marton, N. Blodow, and M. Beetz. Learning Informative Point Classes for the Acquisition of Object Model Maps. In *Proceedings of the 10th International Conference on Control, Automation, Robotics and Vision (ICARCV), Hanoi, Vietnam, December 17-20*, 2008. 1

[13] W. Wohlkinger and M. Vincze. 3d object classification for mobile robots in home-environments using web-data. *19th International Workshop on Robotics in Alpe-Adria-Danube Region RAAD*, 2010. 1

[14] Z. Zhang. Iterative point matching for registration of free-form curves. 1992. 5

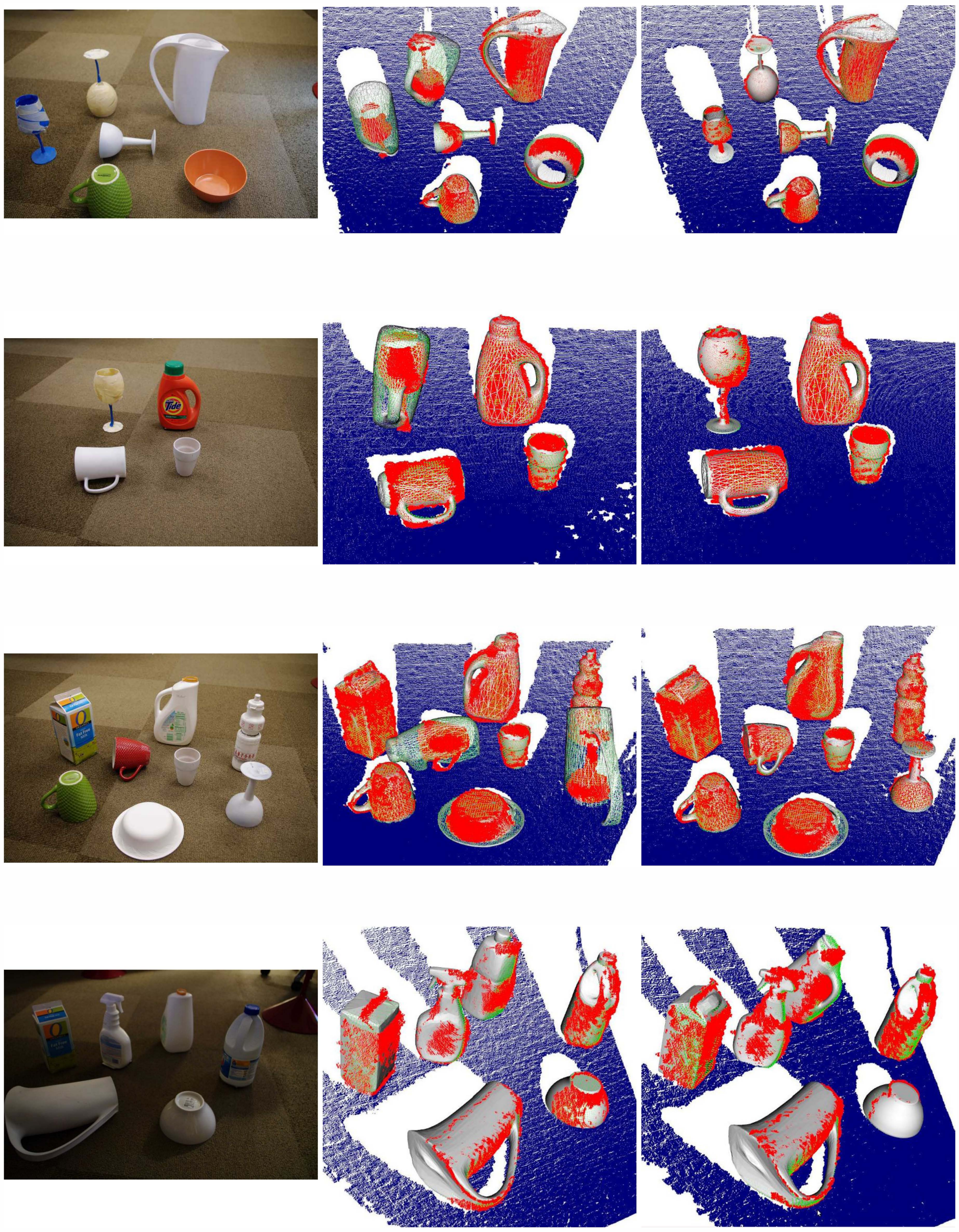

Figure 7. First column: Image of the scene, second column: results obtained using VFH and the third column using CVFH. Both VFH and CVFH results include the camera's roll histogram and the post-processing step to refine results.