Adding Rotational Robustness to the Surface-Approximation Polynomials Descriptor

Richard Bormann, Jan Fischer, Georg Arbeiter and Alexander Verl
Fraunhofer IPA, 70569 Stuttgart, Germany
{richard.bormann, jan.fischer, georg.arbeiter, alexander.verl}@ipa.fraunhofer.de

***Abstract*—The Surface-Approximation Polynomials (SAP) descriptor has been shown to be an appropriate global surface descriptor for object categorization tasks in robotic applications [1]. Nevertheless, in the original formulation the SAP descriptor is not invariant against rotations around the camera axis. This paper explains and evaluates two methods which pre-process the input data to yield repeatably well-aligned point clouds for the computation of the SAP descriptor. We show that the SAP descriptor can be rendered robust against rotations while retaining almost the full performance of the original approach which is superior to GFPFH, GRSD and VFH.**

I. Introduction

Understanding the elements of the environment is essential for robots that are supposed to assist humans in their homes. Only if robots are able to recognize objects in their surroundings, they can manipulate them in a useful way. However, the large variety of objects in home environments turns instance-based object recognition infeasible as the appearance of each single object would have to be learned individually by the robot. Object categorization instead strives to recognize object classes. Hence, objects of a known class can still be recognized even if a certain instance is completely new to the robot. Moreover, the recognition problem becomes more tractable since there are less classes than individual objects.

The Surface-Approximation Polynomials (SAP) descriptor has been recently introduced as a global 3D surface descriptor that is well-suited for the task of object categorization with a robot [1]. The SAP descriptor approximates the surface geometry of a single-shot view onto an object with polynomials. The categorization system based on the SAP descriptor can determine the category label of unknown objects that are captured with a depth sensing device like a PMD CamCube or a Microsoft Kinect. It has been shown in [1] that the SAP descriptor is robust enough to compensate smaller viewpoint changes up to 15° in pan and tilt direction. However, roll rotations of the object or camera cannot be handled at all with the basic approach. Especially, modeling all possible roll rotations with sufficiently many training views is infeasible as the number of required images would explode. Please consult Fig. 1 for the definitions of rotations.

In the real world objects may occur in any arbitrary pose. Consequently, the SAP descriptor should be able to cover every object pose. In this paper we propose and carefully evaluate two methods which align the input data canonically: a full 6 DOF transformation based on Principal Component Analysis (PCA) over the input point cloud as well as a roll compensation which only aligns the input data to a common roll angle. We furthermore introduce a rule to obtain a repeatable definition of the axis directions of PCA.

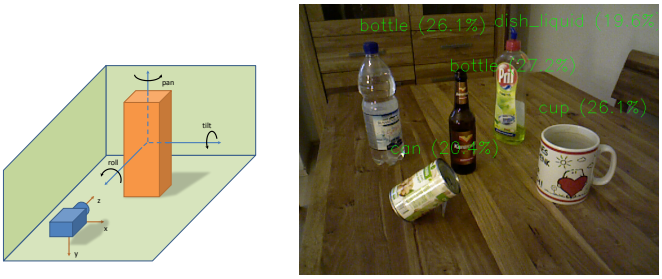


Fig. 1. Definition of the rotational axes for the analysis of the rotational robustness of the SAP descriptor and an example image with real categorization results of variously aligned, previously unseen objects.

The outline of the paper is as follows. In Section II we discuss relevant work to the topics of object categorization and pose alignment. Section III explains the employed approaches, which are evaluated in Section IV. We conclude in Section V with a summary and an outlook for future work.

II. Related Work

Object categorization is a topic of high interest in robotics. The most popular global descriptors that can be computed fast enough for using them in robotics are Global Fast Point Feature Histograms (GFPFH) [2], Global Radius-based Surface Descriptors (GRSD) [3], and Viewpoint Feature Histogram (VFH) [4]. GFPFH builds histograms on local Fast Point Feature Histograms [5] which themselves are histograms on the relative pose of local coordinate frames determined at all point pairs within a neighborhood. The GRSD descriptor is composed similarly to the GFPFH descriptor from local RSD features, which basically represent the local minimum and maximum curvature around a point. VFH is very similar to GFPFH but supposed to also encode the viewpoint at the visible object surface. VFH includes the camera axis in the computation of FPFH histograms to establish viewpoint dependent signatures for the trained objects. The recently proposed SAP descriptor [1] instead directly builds a global descriptor without computing local features and produces categorization results superior to the previous descriptors. We will provide a short description of the SAP descriptor in Section III-B.

in: Humanoids 2012 : 12th Int. IEEE-RAS Conference on Humanoid Robots; November 29-December 1, 2012, Osaka, Japan; S. 409-416

As stated above, the problem of the original SAP descriptor is the missing inherent invariance to roll rotations. Pose normalization of 3D object models is an important topic in the shape retrieval literature where it is applied to transform objects into a canonical orientation w.r.t. translation, size and rotation for the use with pose variant descriptors. The most popular approach in this community seems to be a PCA-based alignment [6]–[11] because of its simple and fast computation and numerical robustness. However, a serious problem with PCA is the repeatable definition of the coordinate axis into positive or negative direction of the principal axes. In [6] all four possible configurations were tested and the orientation with the best similarity between two query objects was chosen. However, our task does not involve two previously known objects. Therefore, we define the axis definitions according to the distribution of points of the query object in the new coordinate system. This method is similar to the approach of [12] where the axes are directed to the side with a greater total area of the polygons. In [8] continuous PCA is introduced to deal with different triangle resolutions in polygon meshes. These problems do not occur with volumetric or mass-based 3D models as we use.

A second classical method for pose alignment is Extended Gaussian Images (EGI) [13]. This algorithm computes the projections of the surface normals on a Gaussian sphere around the object. In [10] maximum normal distribution is proposed as another normal-based pose alignment method for polygon meshes. The idea is to create a histogram over the total area of surfaces which have the same distance to the object center and the same surface normal. Then the normal direction with the largest total area is picked as first principal axis and the orthogonal normal with next largest area as second. Since our input data does not contain meshes we use a PCA-based full pose alignment with adequate axis definitions and a roll compensation with PCA involved in the computations.

III. Methods

Besides the detailed description of the orientation alignment this section briefly summarizes the principle of the SAP descriptor and the underlying categorization framework. The next paragraph starts with a description of data preparation.

A. Data Acquisition and Segmentation

The SAP descriptor is a global descriptor which describes the surface of objects. Therefore, segmented object data is needed to compute the SAP descriptor. After capturing a depth image the scene is segmented in three steps. First, the amount of points in the input point cloud is reduced with a voxel filter that has a leaf size of 7.5 mm. Then the larger planes are iteratively estimated and removed from the input point cloud. Third, the remainder of points is aggregated with Euclidean clustering. Those clusters which contain more than 50 points are then considered as object candidates and forwarded to the SAP descriptor computation. The functions for clustering base upon the PCL library [14].

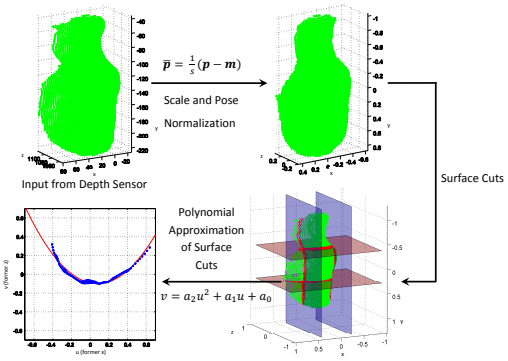


Fig. 2. Computation scheme of the SAP descriptor. The upper left image shows the raw point cloud input. Following the arrows, pose and scale normalization is applied, surface cuts are extracted (red and blue planes cut the surface, cuts indicated as red points) and finally approximated with a polynomial (original points in blue, the red line shows the polynomial).

B. The Surface-Approximation Polynomials Descriptor

The Surface-Approximation Polynomials (SAP) descriptor has been described in detail in [1]. Therefore, we only provide a schematic summary of the algorithm at this place. The basic idea behind the SAP descriptor is to represent object classes by the shape of their surface. As shown in Fig. 2 this is accomplished by normalizing the input point cloud $\mathcal{P}$ to a common centroid and scale, cutting the surface with planes perpendicular to the camera plane and approximating the geometry of the cuts with polynomials via linear regression. Having n_x cuts along the x-direction of the camera plane and n_y cuts along the y-axis this yields n_x+n_y parameter vectors ${}^{\mathbf{i}}\mathbf{a}^{\mathrm{T}}, i = 1, \ldots, n_x + n_y$, of the polynomial coefficients. Furthermore, we compute a Principal Component Analysis (PCA) to obtain the eigenvalues $\lambda_1, \lambda_2, \lambda_3$ which serve as a measure of object size within the three principal directions. The SAP descriptor is a concatenation of these three size parameters and the polynomial coefficients

$$\mathbf{c} = \left[\frac{\lambda_1}{\gamma}, \frac{\lambda_2}{\lambda_1}, \frac{\lambda_3}{\lambda_1}, {}^{\mathbf{1}}\mathbf{a}^{\mathrm{T}}, \ldots, {}^{\mathbf{n_x+n_y}}\mathbf{a}^{\mathrm{T}} \right]. \quad (1)$$

To support a range of object sizes λ_2 and λ_3 contribute only with their relation to λ_1. λ_1 is stored with an optional scale parameter γ to incorporate one measure of absolute size.

C. Extensions for Rotation Invariance

The unaligned SAP descriptor as described in [1] is only invariant against translation and scale but not against rotations, especially around the camera axis (roll). Although it is possible to model viewpoints from different pan or tilt angles with respective training images from a grid around the object, there is no way to capture different poses in roll direction without capturing a vast mass of images. To be able to handle objects in arbitrary poses, rotation invariance has to be accomplished by further measures. Here we propose a full pose alignment with PCA that can compensate pan, tilt and roll rotations of the captured objects as well as a roll compensation method which still has a need for sufficient coverage of training views regarding pan and tilt rotations.

in: Humanoids 2012 : 12th Int. IEEE-RAS Conference on Humanoid Robots; November 29-December 1, 2012, Osaka, Japan; S. 409-416

1) PCA-based Pose Normalization: To receive a repeatable, scale- and rotation-invariant description, the pose of the point cloud is normalized by computing the mean $\mathbf{m}$ and the principal axes $\mathbf{v_1}, \mathbf{v_2}, \mathbf{v_3}$ via PCA. Every point $\hat{\mathbf{p}}$ of the camera coordinate system $\hat{\mathfrak{C}}$ with the axes $\hat{\mathbf{x}} = (1,0,0), \hat{\mathbf{y}} = (0,1,0)$, and $\hat{\mathbf{z}} = (0,0,1)$ is then translated to shift the point cloud's center into the origin, rotated such that the eigenvectors are aligned with the coordinate axes and scaled with the largest eigenvalue λ_1 yielding the normalized point

$$\mathbf{p} = \frac{1}{2\sqrt{\lambda_1}} \cdot \begin{bmatrix} \mathbf{v_1} & \mathbf{v_2} & \mathbf{v_3} \end{bmatrix}^{\mathrm{T}} \cdot (\hat{\mathbf{p}} - \mathbf{m}) \quad . \tag{2}$$

Translating the center of the point cloud to the origin ensures translation invariance w.r.t. the coordinate system of the depth sensor while the rotation compensates for any object rotation around the camera axis and for minor rotations around the other two axes. The scaling operation effects that the majority of the coordinates resides in the range of [-1,1].

As the sign of the direction of the eigenvectors obtained from PCA does not necessarily coincide between several recordings, we have to enforce a repeatable orientation of the new coordinate system $\mathfrak{C}$ with the coordinate axes $\mathbf{x} = \mathbf{v_1}, \mathbf{y} = \mathbf{v_2}$, and $\mathbf{z} = \mathbf{v_3}$. Therefore, we first check that the eigenvectors constitute a right-handed system which is the case if the triple product

$$(\mathbf{v_1} \times \mathbf{v_2}) \cdot \mathbf{v_3} > 0 \tag{3}$$

is positive. If condition (3) is not met, we invert the coordinates of eigenvector $\mathbf{v_3}$ before transforming the point cloud. Then, we obtain a repeatable coordinate system if the following three rules are fulfilled:

1) The new z-axis, which has the coordinates $\mathbf{z} = \mathbf{v_3}$, must point towards the camera. Hence, the condition $\hat{\mathbf{z}} \cdot \mathbf{z} < 0$ must hold since the initial $\hat{\mathbf{z}}$-axis of the camera coordinate system with coordinates $\hat{\mathbf{z}} = (0,0,1)^{\mathrm{T}}$ points away from the camera.
2) The majority of points should have negative x-coordinates in the new coordinate system $\mathfrak{C}$.
3) The new coordinate system $\mathfrak{C}$ is a right-handed system.

These conditions are checked in the given order. If rule 1 is not fulfilled, we only change the signs of eigenvectors $\mathbf{v_2}$ and $\mathbf{v_3}$ before transforming the point cloud to keep the coordinate system right-handed. The second condition can only be verified after the transformation of the points. If it is not met, we have to negate the eigenvectors $\mathbf{v_1}$ and $\mathbf{v_2}$ and the x- and y-components of the transformed points to keep the coordinate system right-handed at the same time. After executing the preceding steps, rule 3 is already fulfilled. Rule 3 is always enforced in step 1 and step 2 by negating $\mathbf{v_2}$, the $\mathbf{y}$-axis, and the y-coordinates. After the verification of all three rules, the eigenvectors $\mathbf{v_1}, \mathbf{v_2}$, and $\mathbf{v_3}$ correspond to the new coordinate axis $\mathbf{x}, \mathbf{y}$, and $\mathbf{z}$, respectively. After normalization, the surface of the object is aligned in a way that the two dimensions with the largest extent correspond to the x- and y-axes. We evaluate the success of this measure in Sec. IV-B.

2) Roll Compensation: The second approach to render the SAP descriptor rotation invariant w.r.t. roll rotations does not apply a full 3D transform to the point cloud but only aligns its rotation around the camera axis. This way, roll rotations of objects are made transparent to the algorithm. The roll compensation is motivated by the possible misalignments with full PCA (see Sec. IV-B) and was developed with the goal to transform the point cloud as little as possible.

The roll compensation algorithm works as follows: first a silhouette image is created from the projection of the point cloud onto the camera plane. Then we compute the centroid $\mathbf{m}$ of this 2D silhouette as well as the two principal axes $\mathbf{v_1}$ and $\mathbf{v_2}$ using PCA. Next the silhouette is rotated to be aligned with the principal axes and it is counted whether more points have positive x-coordinates than negative. If this condition does not hold, the directions of the principal axes are negated. This step ensures to have a repeatable definition of the direction of the new coordinate system. Then we compute the angle α between the first principal axis $\mathbf{v_1}$ and the image's x-axis $(1,0)$:

$$\cos\alpha = \frac{1}{\sqrt{v_{11}^2 + v_{12}^2}} \cdot \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad . \tag{4}$$

Finally, we rotate every point $\hat{\mathbf{p}}$ of the original point cloud $\mathcal{P}$ by angle α around the camera axis $\hat{\mathbf{z}}$:

$$\mathbf{p} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} (\hat{\mathbf{p}} - \mathbf{m}) + \mathbf{m} \quad . \tag{5}$$

This yields a repeatable roll compensation for the input point cloud so that the SAP descriptor can then be computed on a point cloud with aligned roll angle. Section IV-C examines how well the roll compensation works in practice.

D. Classification Framework

The object categorization framework is identical to [1]. It is supposed to serve two purposes: first the system should be able to search for instances of a certain class and assert whether test objects belong to it. This is a binary classification task. Second, it should be able to label unknown objects with the correct class. This is a multi-class classification problem. To be able to deal with both problems the classification framework builds on binary Random Forest classifiers [15] which separate each class against the remainder of classes. Binary decisions are obtained by directly querying these classifiers. A probabilistic multi-class extension is employed for the labeling task, which directly computes the labels from the likelihoods of the binary classifiers and their decision reliabilities that originate from statistics.

IV. Evaluation

The SAP descriptor of unaligned point clouds has already been examined in [1]. In this paper we discuss the impact of the PCA-based pose alignment and roll compensation and compare the outcomes with those from unaligned input data. All results reported on database tests are determined with a 10-fold leave out one object cross-validation.

A. Database

Database tests are conducted with the database of household objects named IPA-2 in [1]. It contains 151 objects from 14 classes. Among these classes are binders, bottles, cans, cups, dishes, drink cartons, computer mice, pens, silverware, etc. Each object was captured 36 times with a PMD CamCube from a light tilt angle with an offset of 10° in the pan angle. The average number of points per object is 26491. A detailed description of the IPA-2 object database can be found in [16]. This set is publicly available at http://www.kyb.mpg.de/nc/employee/details/browatbn.html.

B. PCA-based 6 DOF Pose Normalization

This section analyzes the robustness of the SAP descriptor against rotations and camera distance of the captured objects when the input point cloud is aligned with the PCA-based approach (Sec. III-C1). The analysis of the basic SAP descriptor in [1] shows that the SAP-7-7-2 configuration yields very good results. Thus, all experiments in this section will be conducted with this parameter setting if not mentioned else. The naming scheme for SAP descriptors is SAP-n_x-n_y-n_p, where n_x and n_y describe the number of cuts along the x- and y-coordinate axes (after alignment of the point cloud) and n_p denotes the degree of the approximating polynomial.

1) Theoretical Analysis: The function and power of PCA-based pose normalization is demonstrated on a cuboid. Fig. 3(a) displays this cuboid as well as a multitude of camera view points which pan in the range $[5.625^\circ, 84.375^\circ]$, tilt within $[15^\circ, 75^\circ]$ and are depicted as black dots with a black line indicating the camera axes. The black point in the middle of the object is the real centroid of the cuboid whereas the red points with the coordinate frames attached display the object centers that are computed from the three visible surfaces of the cuboid. The offset between the centroid that we can estimate from the visible data and the real centroid has an effect on the chosen translation compensation since the position of the estimated centroid depends on the view point. The locations of the estimated centroids differ since the depth sensor samples the less points from a surface the more the viewing angle onto the surface becomes acute. Let $\mathcal{S} = \{S_1, S_2, S_3\}$ denote the set of visible surfaces of the cuboid. Then the theoretical centroid $\mathbf{x_s}$ of the visible surfaces is computed as

$$\mathbf{x_s} = \mathbf{x}(S_1)A(S_1) + \mathbf{x}(S_2)A(S_2) + \mathbf{x}(S_3)A(S_3) \quad (6)$$

where $A(S)$ stands for the area of surface S and $\mathbf{x}(S)$ for the centroid of S. However, depending on the viewing angle the depth sensor can only capture a ratio of the maximum amount of points that could be captured from a surface if the camera axis was perpendicular to the surface. We model this effect with the following ratios for the visible portions of each area where α is the pan angle and β represents the tilt angle:

$$\begin{aligned} S_1&: \quad \cos(\alpha)\cos(\beta), \\ S_2&: \quad \sin(\alpha)\cos(\beta), \\ S_3&: \quad \sin(\beta)\ . \end{aligned}$$

The view-dependent centroids (red points in Fig. 3(a)) are computed according to Eq. (6) where every area $A(S_i)$ only accounts with the respective view-dependent ratio. It shows that the perceived centroids still lie quite close to each other when the change in viewing angle is below 15°. If the surface of the object is sufficiently smooth this small translation of the centroid will not affect the polynomial approximation substantially given that the rotation can be compensated.

The rotation compensation is supposed to be accomplished by the PCA-based alignment. The idea is to determine the principal axes of the captured object, which are supposed to be stable under minor rotations, and rotate the surface to be aligned with these principal axes. While computing the principal axes we obey the aforementioned ratios of visible points on the object's surfaces to obtain a realistic result. The pose normalized coordinate system is assigned to the principal axes in descending order of corresponding eigenvalues, that is the new x-axis is the eigenvector with the largest eigenvalue. The resulting pose normalized coordinate axes for the cuboid example are displayed for all viewing angles at the position of the estimated centroids in Fig. 3(a). The red axis displays the x-axis, the y-axis is green and the z-axis is blue. It is visible that the estimated principal axes correspond roughly to the real principal axes of the cuboid and all coordinate frames are similarly aligned over a wide range of view points. To illustrate the latter fact, a comparison of the distribution of coordinate frames without and with PCA-based pose normalization is provided in Fig. 3(b) and Fig. 3(c), respectively. While the original coordinate frames scatter a lot, the normalized coordinate frames have little deviation over intermediate viewpoint changes and barely follow the camera movements. Consequently, PCA-based pose normalization will align object surfaces similarly within an intermediate range of pan and tilt rotations and hence yield similar SAP descriptors. Roll rotations are not considered in this analysis because the PCA-based pose normalization and the computation of repeatable axis directions yield the same normalized pose for every roll angle while pan and tilt angles are fixed.

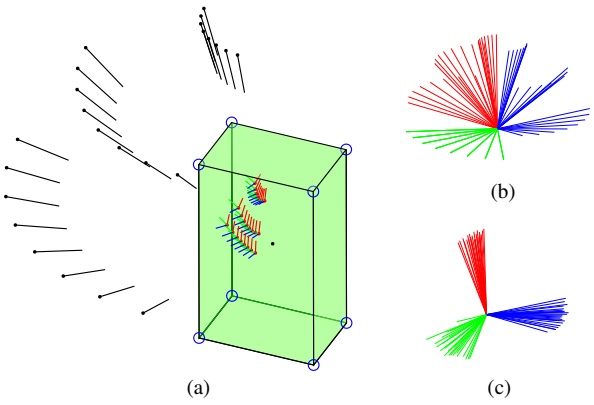


Fig. 3. (a) The normalized coordinate frames are displayed at the estimated centroids for the considered viewing angles, which are displayed as black dots and lines. Collection of (b) the original coordinate frames and (c) the normalized coordinate frames of the cuboid seen from those viewpoints.

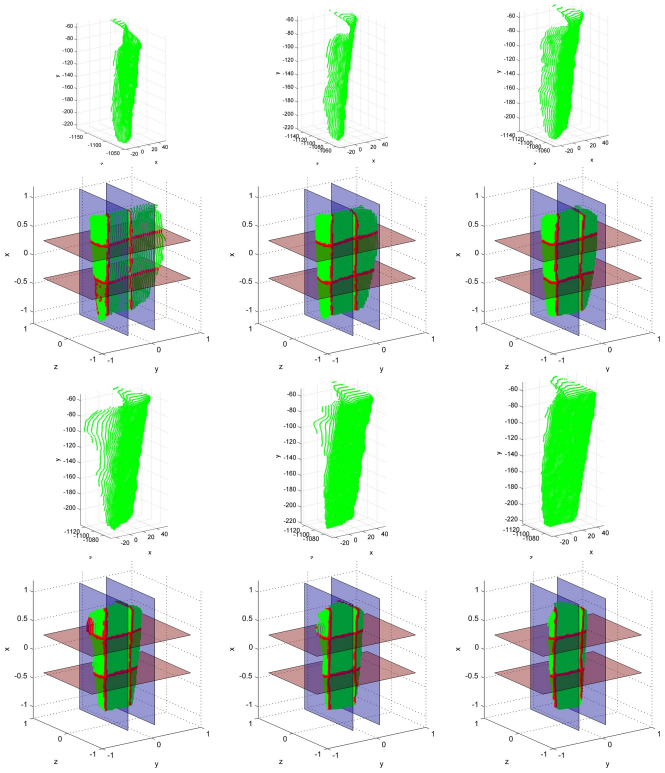


Fig. 4. Point cloud of a milk box captured from six neighboring viewing angles. The first and the third row show the original data from the depth sensor. Row two and four display the corresponding point clouds which are aligned with PCA-based pose normalization. Two exemplary surface cuts are drawn in each cutting direction into the pose normalized views.

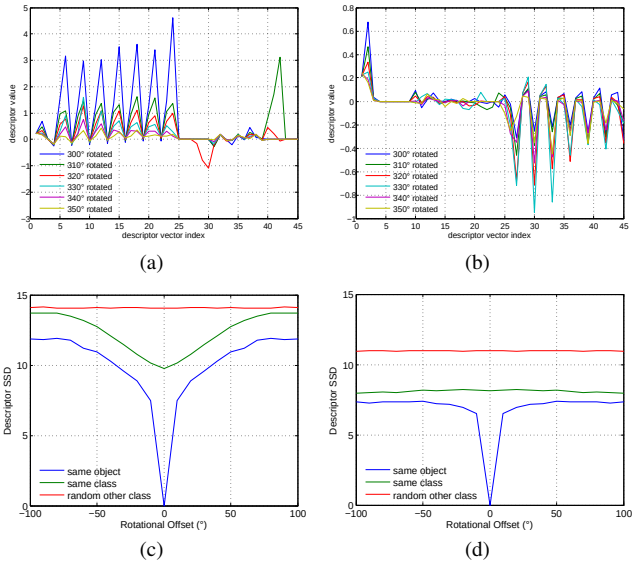


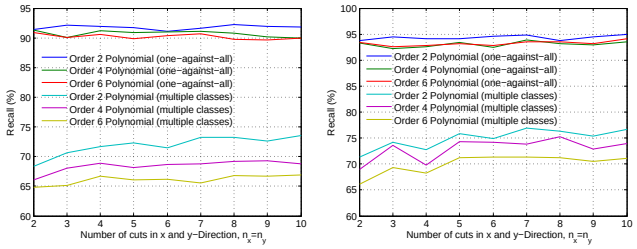
Fig. 5. Analysis of the similarity of SAP descriptors: SAP-7-7-2 descriptors from (a) the original snapshots and (b) the pose normalized point clouds of the example views of the milk box. Averaged descriptor similarity between views with varying angular offset on (c) the original data and (d) the pose normalized data. Average similarity is measured against rotations of the same object, objects from the same class and objects of other classes.

To demonstrate the effect of PCA-based pose normalization on real data Fig. 4 shows a sequence of views onto a milk box. This box rotates on a rotary disc so that the camera movement is effectively a pan rotation with an angular offset of $10°$ between successive views. The first and third row show the point clouds as captured by the sensor. The second and fourth row display the corresponding views onto the milk box after PCA-based pose normalization has been applied. While the original point cloud rotates by $50°$ over the sequence the pose normalized views look very similar in all images as predicted by the previous analysis.

To back the claim that the SAP descriptors obtained from pose normalized views are more similar to each other than those obtained from the original views, Fig. 5 provides two pieces of evidence. The first row of images displays the SAP-7-7-2 descriptors of all views of the milk box from Fig. 4. In detail, Fig. 5(a) contains all six SAP-7-7-2 descriptors from the original views whereas Fig. 5(b) displays the SAP-7-7-2 descriptors from the PCA-based pose normalized views. To ensure a fair comparison between both cases the original views are scaled to fit into the unit volume as well. To compare the descriptors of both approaches please consider that the axis definitions change through the pose normalization. In the milk box example, the x- and y-axis definitions swap between original and normalized view and consequently, the SAP polynomial coefficients from the first half of one diagram can be found in the second half of the other diagram. The corresponding coefficients are inverted because the z-axis direction switches through the pose normalization. Only the first three size components of the descriptors correspond in both cases and take the same values. We can observe that the SAP-7-7-2 descriptors from a range of viewing angles of $50°$ do not differ much when PCA-based pose normalization is applied whereas the descriptors obtained from the original views exhibit a transition in the coefficients in the first half of the descriptor. This steady decrease in magnitude is caused by the pan rotation of the object which lets the surface appear as a backwardly slanted plane at first and transitions to a plane parallel to the camera in the end. The pose normalized views present a parallel plane for all views instead which results in very similar descriptor coefficients in all cases.

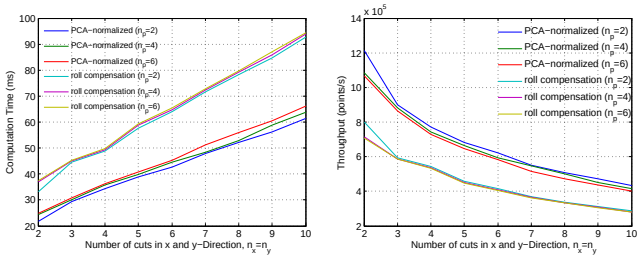
To show that this fact holds in general this analysis has to be extended to the whole database. Figure 5 therefore displays two plots in the second row in which the similarity of SAP-7-7-2 descriptors of neighboring views is studied on the whole dataset. Similarity between two descriptors c_1 and c_2 is measured as the sum of squared differences (SSD) $SSD = \|c_1 - c_2\|^2_{L_2}$. We can see in Fig. 5(c) that the descriptor similarity decreases significantly with growing offset between two views if the descriptors are computed on the original point cloud. If computed on the normalized data instead (see Fig. 5(d)), the similarity of descriptors from neighboring views barely increases even for larger rotations. This finding indicates that PCA-based pose normalization helps to keep SAP descriptors computed from neighboring views quite similar.

in: Humanoids 2012 : 12th Int. IEEE-RAS Conference on Humanoid Robots; November 29-December 1, 2012, Osaka, Japan; S. 409-416



(a) (b)

Fig. 6. Comparison of different configurations of the SAP descriptor with varying numbers of surface cuts and degrees of the polynomials. The input data was aligned with (a) PCA and (b) roll compensation.



(a) (b)

Fig. 7. Dependency of (a) computation time and (b) throughput of the SAP descriptor for increasing numbers of surface cuts and polynomial orders. The input data is either aligned with PCA or with roll compensation.

Although the preceding analysis has pointed out several desirable properties of the chosen PCA-based pose normalization it is well known that this kind of pose normalization is problematic and possibly unstable if applied to objects which do not have such canonical orientations as the cuboid [17]. Therefore, we test the impact of pose normalization by measuring the performance on the object categorization task.

2) Database Tests: According to the analysis in [1] the influence of the numbers of surface cuts n_x and n_y as well as the degree of the approximating polynomials needs to be examined. Figure 6(a) displays the recall rates for the binary classification problem of separating one object class against the others as well as for the multi-class labeling task where each object view has to be assigned one of the class labels. The influence of the number of cuts is as expected and coincides with findings of experiments with the unaligned data: especially in the multi-class problem the recall rates increase steadily with growing numbers of surface cuts. The binary categorization performance, however, remains almost constant independent of the number of surface cuts. Nevertheless, the increasing performance for the multi-class task indicates that the binary decisions become more confident, that is the probabilities for the respective decisions of the binary classifiers grow with the number of surface cuts.

As in the unaligned case approximations of higher order polynomials yield a worse performance. Manual inspection of the descriptors provides an explanation for this observation: it shows that higher order polynomials are less stable and tend to model the noise from the sensor. Besides these qualitative observations we also confirm that the SAP-7-7-2 configuration proves to be among the top performers, however, with slightly lower recall rates than with unaligned data. In the binary classification case the performance drops from 94.9% with unaligned data to 91.7% with PCA-aligned data and for the multi-class labeling task the performance decreases from 77.9% to 73.2%. The good performamce in the unaligned case is not surprising since the objects in the database are already well-aligned. The decrease by almost 5% of multi-class recall indicates that the PCA alignment introduces a significant number of misalignments.

Next, Figure 7(a) shows the average computation time for one SAP descriptor and Figure 7(b) the respective computational throughput. These results resemble those of the unaligned computation very much in qualitative and quantitative aspects. Thus, the additional pose alignment does not introduce significant overhead for the computation. There is a linear increase in computation time with rising numbers of surface cuts. The computation time for the SAP descriptor is quite low as all examined configurations are determined within less than 100 ms on one core of a 2.8GHz Intel I7 mobile processor with 6GB RAM. The runtime of the SAP-7-7-2 configuration e.g. allows to compute the descriptor with almost 21 Hz. That is SAP could classify up to 21 objects in a scene within one second which is a respectable rate.

C. Roll Compensation

We will not provide a similarly extensive evaluation for this approach of orientation compensation as the results strongly resemble those of the unaligned approach. The reason for this lies in the good alignment of objects in the database which renders the input data almost equal for both database tests. However, the success of roll compensation on objects in other poses than those in the training data is proven by the examples in Fig. 1 and 9, e.g. for cans, cups and the binder. The following analysis indicates the equalities and differences to the results of the unaligned approach.

The impact of parameters n_x, n_y and n_p on the categorization performance is shown in Fig. 6(b). The qualitative results correspond with previous findings and the recall rate of 77.0% of the SAP-7-7-2 descriptor with roll compensation comes close to the 77.9% of the unaligned method. The computation times with roll compensation are higher than with PCA-based alignment or without alignment by 20 ms to 30 ms as we can see in Fig. 7(a). Nevertheless, the SAP-7-7-2 configuration still classifies almost 14 objects per second. The linear dependency on the number of cuts remains.

D. Comparison of the Approaches

This paragraph compares the unaligned, PCA-aligned and roll-compensated SAP descriptors according to their categorization performance, runtime and robustness against rotations and scale changes. Table I summarizes the categorization performance and computation times of the three variants of SAP descriptors and other descriptors from literature. It shows that the SAP variant with roll compensation achieves

TABLE I

COMPARISON OF SEVERAL DESCRIPTORS REGARDING MULTI-CLASS CATEGORIZATION PERFORMANCE, AVERAGE COMPUTATION TIME PER VIEW AND AVERAGE THROUGHPUT IN POINTS PER SECOND.

Descriptor	Recall	Time	Throughput
Shape Distributions [16]	25.4 %	31 ms	$\sim$ 855 000 pts/s
Shape Index [16]	34.6 %	78 ms	$\sim$ 339 000 pts/s
Shape Context 3D [16]	55.2 %	234 ms	$\sim$ 113 000 pts/s
Depth Buffer [16]	72.9 %	**16 ms**	$\sim$ **1 656 000 pts/s**
GFPFH [1]	54.4 %	921 ms	28 928 pts/s
GRSD [1]	56.1 %	957 ms	27 841 pts/s
VFH [1]	68.4 %	93 ms	205 883 pts/s
SAP-7-7-2			
unaligned [1]	**77.9 %**	57 ms	463 439 pts/s
with roll compensation	77.0 %	72 ms	370 314 pts/s
with PCA alignment	73.2 %	48 ms	552 262 pts/s

almost the recall rate of the unaligned SAP descriptor but needs 15 ms of additional computation time. The similar recall rate indicates that roll compensation works with few errors since the performance of the unaligned SAP descriptor is kind of a limit for methods with pose alignment as the database objects are already well-aligned. The recall rate of the SAP descriptor with PCA alignment is almost 5% lower than this limit suggesting that some misalignments occur. The faster computation time compared to the unaligned method is caused by the changed orientation which affects the number of points on surface cuts. All runtime measurements were taken on one core of a mobile I7 2.8 GHz machine with 6GB RAM. A confusion matrix for the categorization with roll compensation is provided in Fig. 8(d). Many class labels are found quite reliably whereas the occurring confusions can usually be easily explained, e.g. bottles and dishliquids have a similar shape and silverware and scissors look the same when seen from the slim side.

The next analysis evaluates the robustness of the variants of SAP descriptors against rotations of the object in pan and tilt direction. For the evaluation on pan rotations we just exclude the respective views from the training data to yield sparser object models sampled only every α degrees in the pan direction. Fig. 8(a) reports on the recall rates obtained with respect to the angular offset $\alpha/2$ of the views of unknown test objects. This means, the angles reported in the diagram correspond with the maximal angular offset to the closest view on another object of this class available in the training set. It is remarkable that in all three cases the performance is still around 60% when the training data only consists of 4 views of each object. We also notice that the recall rate virtually remains constant up to an offset of 15° for the PCA-aligned SAP descriptor and up to 10° for the other two variants. The gap between the PCA-aligned and the unaligned descriptor remains almost constant over the whole range and is surprising as the pose alignment should be benefitial with few views. Apparently the misalignment rate of the PCA-based approach eats up this potential advantage. The performance of roll compensation begins at the same level as the unaligned approach but degrades with growing angular offset towards the performance of PCA alignment. A similar analysis has been carried out for tilt rotations.

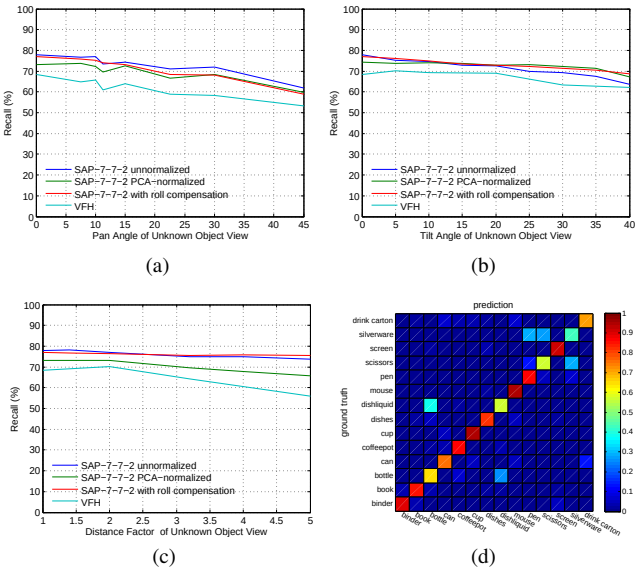


Fig. 8. Robustness of the three variants of the SAP descriptor with respect to (a) pan and (b) tilt rotations as well as (c) camera distance. (d) Confusion matrix for categorization with the SAP descriptor and roll compensation.

Lacking real data from all tilt angles, the experimental setup for tilt rotations is different: the original point clouds are tilted by angle β and because of the changed perspective only a ratio of $\cos\beta$ points are kept in the model. The system is trained with data from tilt angle 0° and β. The test data only contains point clouds tilted by $\beta/2$. Fig. 8(b) displays the recall rates with respect to $\beta/2$. Up to tilt angles of 35° the performance keeps above 67% with all three approaches which is quite high. For PCA-based alignment, the recall rate remains almost constant up to that point, for roll compensation it converges to the level of the aforementioned method. For the unaligned data recall decreases steadily and falls below the other methods at tilt offsets of 15°.

The last experiment evaluates the robustness of the three variants of SAP descriptors against varying camera distance to the objects. To emulate different distances between object and camera we downsample the original point clouds randomly to different distance levels, e.g. to simulate the double distance we only keep 25% of the original points. Fig. 8(c) shows the recall rates for various distances. The unaligned and roll compensated SAP descriptors can retain their performance over almost the whole range of analyzed distance factors. The recall rates of PCA-based alignment, however, decrease significantly after the distance doubles. Apparently, the impact of noise in the point measurements grows larger if less points are available and this affects the stability of the PCA-alignment.

The robustness analysis has also been conducted for VFH to allow for a comparison. The rotational robustness is similar to the SAP descriptors but the robustness to camera distance is lower as visible in Fig. 8(a), 8(b) and 8(c).

Finally, we demonstrate the categorization system on real scenes with previously unseen objects in Fig. 1 and Fig. 9. We selected the roll compensation approach for point cloud

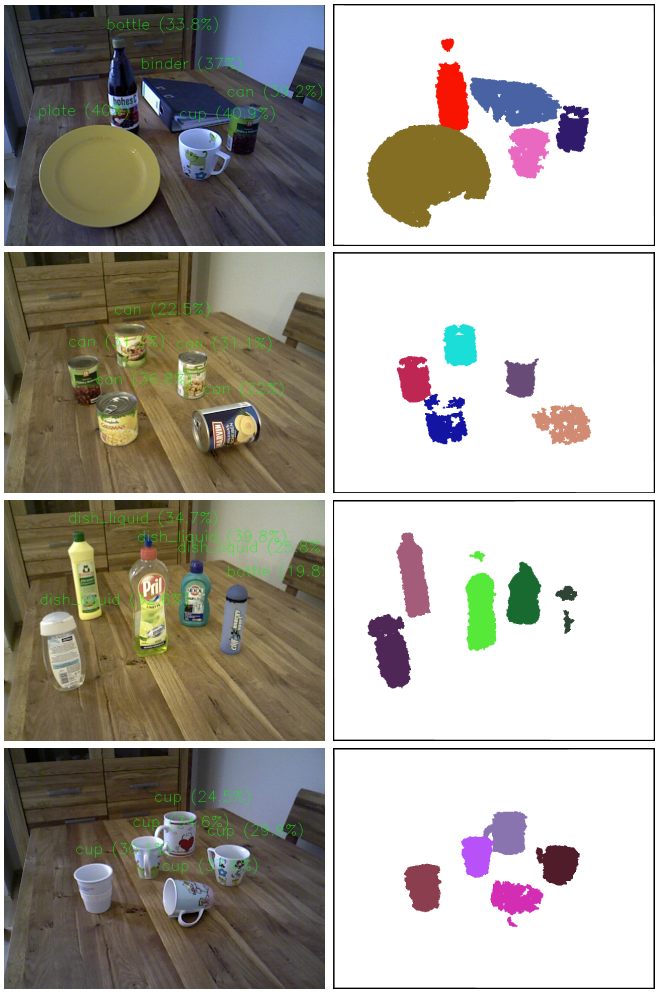


Fig. 9. Exemplary real world scenes with objects from various classes in diverse poses. The objects are not part of the training set. The point clouds are aligned with roll compensation before the SAP descriptor is computed. The right column shows the corresponding object clusters of the point cloud.

alignment to benefit from the higher recall rates and the good robustness against transformations including roll rotations, which are not covered by the unaligned SAP descriptor. We placed the objects in different distances to the camera and turned them in various pan, tilt and roll directions. The recognized object classes are denoted on top of each object with the probability mass for this label in brackets. Although the probability is only in the range of 20% for several objects the alternatives often have significantly lower probabilities. Please notice that a probability of 50% means that no other object can be more likely. Consequently, probabilities of 40% are already strong assertions.

V. Conclusions and Outlook

The analysis of the two proposed pose normalization methods has shown that the PCA-based full pose alignment of the input point cloud is regularly inferior to the approach with roll compensation which can almost achieve the performance of the unaligned SAP descriptor on aligned data. For modeling object classes with the SAP descriptor we therefore recommend to pre-process the input data with roll compensation and capture training images every 20° in pan and tilt direction to obtain optimal performance. If a performance drop up to 5% in the worst case is acceptable, objects can be modeled with 38 views: 12 images per pan rotation at tilt angles -45°, 0°, and 45° as well as one shot from the top and the bottom.

For future research on the SAP descriptor it is planned to substitute the polynomial approximations with splines. Furthermore, we like to add a size parameter to each cut to represent the length of the approximated curves. A transition to part-based models is also planned to cope with occlusions.

VI. Acknowledgments

This research was partly funded from the EU FP7-ICT-287624 Acceptable robotiCs COMPanions for AgeiNg Years.

References

[1] R. Bormann, J. Fischer, G. Arbeiter, and A. Verl, "Efficient Object Categorization with the Surface-Approximation Polynomials Descriptor," in *Spatial Cognition VIII* (C. Stachniss, K. Schill, and D. Uttal, eds.), vol. 7463 of *Lecture Notes in Computer Science*, pp. 34–53, Springer, 2012.

[2] R. B. Rusu, A. Holzbach, M. Beetz, and G. Bradski, "Detecting and segmenting objects for mobile manipulation," in *ICCV*, 2009.

[3] Z.-C. Marton, D. Pangercic, R. B. Rusu, A. Holzbach, and M. Beetz, "Hierarchical object geometric categorization and appearance classification for mobile manipulation," in *Proceedings of the International Conference on Humanoid Robots*, (Nashville, TN, USA), 2010.

[4] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3d recognition and pose using the viewpoint feature histogram," in *Proceedings of the International Conference on Intelligent Robots and Systems*, 2010.

[5] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *2009 IEEE Int. Conf. on Robotics and Automation*, (Piscataway, NJ), pp. 1848–1853, IEEE, 2009.

[6] M. Novotni and R. Klein, "A geometric approach to 3d object comparison," in *International Conference on Shape Modeling and Applications*, pp. 167–175, may 2001.

[7] J. W. H. Tangelder and R. C. Veltkamp, "A survey of content based 3d shape retrieval methods," in *Shape Modeling Int.*, pp. 145–156, 2004.

[8] D. V. Vranic, D. Saupe, and J. Richter, "Tools for 3d-object retrieval: Karhunen-loeve transform and spherical harmonics," in *IEEE 2001 Workshop Multimedia Signal Processing, Cannes, France*, 2001.

[9] D. V. Vranic, "An improvement of rotation invariant 3d-shape based on functions on concentric spheres," in *ICIP (3)*, pp. 757–760, 2003.

[10] J. Pu and K. Ramani, "A 3d model retrieval method using 2d freehand sketches," in *Proceedings of the 5th International Conference on Computational Science - Volume Part II*, pp. 343–346, 2005.

[11] R. Ohbuchi, K. Osada, T. Furuya, and T. Banno, "Salient local visual features for shape-based 3d model retrieval," in *Shape Modeling International*, 2008.

[12] M. Elad, A. Tal, and S. Ar, "Content based retrieval of VRML objects - an iterative and interactive approach," in *Proc. Eurographics multimedia workshop*, pp. 97–108, 2001.

[13] B. K. P. Horn, "Extended gaussian images," *Proceedings of the IEEE*, vol. 72, no. 2, pp. 1671–1686, 1984.

[14] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *Proc. of Int. Conference on Robotics and Automation*, 2011.

[15] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[16] B. Browatzki, J. Fischer, B. Graf, H. Bülthoff, and C. Wallraven, "Going into depth: Evaluating 2d and 3d cues for object classification on a new, large-scale object dataset," in *Proc. of Int. Conf. Computer Vision Workshop on CD4CV*, pp. 1–7, 2011.

[17] J. Pu, L. Yi, X. Guyu, Z. Hongbin, L. Weibin, and Y. Uehara, "3d model retrieval based on 2d slice similarity measurements," in *Proc. of 3D Data Processing, Visualization, Transmission*, pp. 95–101, 2004.