


# Improving object orientation estimates by considering multiple viewpoints

## Orientation histograms of symmetries and measurement models for view selection

Zoltán Csaba Márton<sup>1</sup>  · Serkan Türker<sup>2</sup> · Christian Rink<sup>1</sup> · Manuel Brucker<sup>1</sup> · Simon Kriegel<sup>1</sup> · Tim Bodenmüller<sup>1</sup> · Sebastian Riedel<sup>1</sup>

Received: 29 February 2016 / Accepted: 3 April 2017 / Published online: 18 April 2017  
© Springer Science+Business Media New York 2017

**Abstract** This article describes a probabilistic approach for improving the accuracy of general object pose estimation algorithms. We propose a histogram filter variant that uses the exploration capabilities of robots, and supports active perception through a next-best-view proposal algorithm. For the histogram-based fusion method we focus on the orientation of the 6 degrees of freedom (DoF) pose, since the position can be processed with common filtering techniques. The detected orientations of the object, estimated with a pose estimator, are used to update the hypothesis of its actual orientation. We discuss the design of experiments to estimate

the error model of a detection method, and describe a suitable representation of the orientation histograms. This allows us to consider priors about likely object poses or symmetries, and use information gain measures for view selection. The method is validated and compared to alternatives, based on the outputs of different 6 DoF pose estimators, using real-world depth images acquired using different sensors, and on a large synthetic dataset.

## 1 Introduction and motivation

In this work, we explore how histogram filters (Thrun et al. 2005) can improve object pose estimation capabilities, while holding specific benefits for robotic applications.

Human perception is assumed to perform a continuous update of hypotheses about the world, and correcting them through sensory information (Doya et al. 2007). Manipulation tasks, for example, are heavily aided by object recognition, for which visual perception is probably the most important modality (Lynott and Connell 2009). Conversely, perception is itself aided by locomotion and manipulation skills (Grzyb et al. 2012). In the robotics domain as well, interaction and multiple viewpoints (spatio-temporal integration) aid model learning and recognition (Pronobis et al. 2010).

In our previous work, in order to obtain highly accurate object positions and orientations for manipulation tasks, a multi-view 6 DoF pose estimation was integrated into an active perception framework (Kriegel et al. 2013). This article reports on an extension of that work by replacing the simple nearest neighbor clustering method for integrating measurements with a probabilistic one. After evaluating several Bayesian and non-Bayesian approaches for fusing different detections, we propose a histogram filter based

This is one of several papers published in *Autonomous Robots* comprising the Special Issue on Active Perception.

This work has partly been supported by the EC, under contract number H2020-ICT-645403-ROBDREAM.

✉ Zoltán Csaba Márton  
zoltan.marton@dlr.de

Serkan Türker  
serkan.tuerker@navvis.com

Christian Rink  
christian.rink@dlr.de

Manuel Brucker  
manuel.brucker@dlr.de

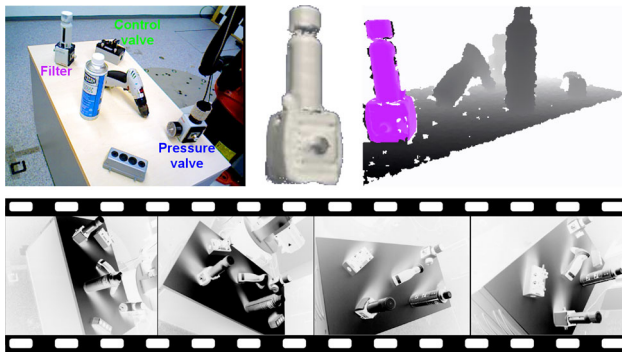
Simon Kriegel  
simon.kriegel@dlr.de

Tim Bodenmüller  
tim.bodenmueller@dlr.de

Sebastian Riedel  
sebastian.riedel@dlr.de

<sup>1</sup> Institute of Robotics and Mechatronics (RM), DLR (German Aerospace Center), Muenchner Strasse 20, 82234 Oberpfaffenhofen-Wessling, Germany

<sup>2</sup> NavVis GmbH, Blumenburgstr. 18, 80636 Munich, Germany



**Fig. 1** Robotic interaction with scenes requires an accurate estimation of object poses, typically achieved through CAD model matching (as shown in the *top row*). Ambiguities can be eliminated and results improved if multiple views are considered (Kriegel et al. 2013) (*bottom*)

method, which performed best in our experiments and enables a viewpoint selection algorithm that further improves detection accuracy.

To exemplify the problem, consider a tabletop scenario (exemplified in Fig. 1) with several objects, as encountered in pick-and-place operations with non-fixed object poses (e.g. shopfloor logistics). The challenge is to estimate the pose for each of the objects, given various problems that affect the pose estimation results:

- objects being outside of view;
- symmetrical structures of the objects;
- systematic and random errors of the algorithm.

The solution to overcome these challenges is to use multiple viewpoints. In such an active exploration, if the pose estimator's detections are correct most of the time, eventually the estimated pose of the object will converge to its actual pose (Kriegel et al. 2013). As the position part of the pose is in the 3D Euclidean space, and misdetections are usually not completely wrong, a simple averaging worked very well in Kriegel et al. (2013). However, to find the optimal orientation, the special orthogonal group  $SO(3)$  has to be considered, with its spherical nature and larger detection ambiguities requiring special attention.

While in our previous work we discussed how to model objects (Kriegel et al. 2013; Marton et al. 2011) and estimated the pose of previously learned objects (Kriegel et al. 2013; Rink et al. 2013; Aldoma et al. 2012; Rink et al. 2016), here the focus is on integrating multiple views and priors about possible errors. We present the principles and the design of experiments for integrating different views, and enable the selection of disambiguating actions.

In the next section we will discuss the most relevant related works, then outline the used pose estimation algorithms in Sect. 3. We discuss the used representation for the orienta-

tions in Sect. 4 and introduce the proposed histogram filtering approach in Sect. 5. Section 6 details the measurement model, estimation of symmetries and viewpoint selection method employed, followed by the results on real and simulated data in Sect. 7 and our conclusions in Sect. 8.

## 2 Related work

In this section we will review the related works of three main aspects of the presented approach. First, pose estimation methods in general are presented, followed by an overview of relevant viewplanning approaches, and finally, the fusion of different detections is discussed.

### 2.1 Pose estimation methods

The proposed method is motivated and derived from histogram filters, but constitutes in fact a voting approach. Most other pose estimation algorithms that are also based on some sort of voting, use local correspondences. In contrast, this approach is based on multiple viewpoints. The general approach in this class of methods is to establish lots of simple correspondences describing (multiple) possible poses of an object in a scene. These represent votes in 6 DoF pose space and dominant clusters of votes are considered probable poses.

Tombari et al. (2012) propose a method, which matches 3D features to attain correspondences, then accumulates evidence of the presence of the object(s) being sought by verifying the consensus of correspondences within a 3D Hough space (Hough 1962). At first, interest points (features) are extracted from both the model and the scene, either by choosing them randomly or by means of a suitable feature detector (Chen and Bhanu 2007; Mian et al. 2010; Novatnack and Nishino 2008). Then each feature point is enhanced with a description of its local neighborhood, i.e., a 3D feature descriptor such as SHOT or FPFH (see survey in Aldoma et al. 2012). Then, given this set of described features extracted from the model and the scene, a set of feature correspondences can be determined and using a matching threshold, unreliable correspondences can be discarded. As the local neighborhood descriptors define coordinate frames, each feature correspondence defines an orientation and votes in the remaining position space for this orientation. If enough features vote for the presence of the object in a given position within the 3D space, then the object is detected and its pose is estimated based on the established correspondences. The method is based on the unique decomposability of rigid motions into a rotation and a translation.

The basic voting approach of Tombari has been proposed earlier by Barequet and Sharir (1994, 1999). Though, they use scalar features and thus a correspondence does not vote for a single orientation, but a set of orientations. Rink et al. (2013) reformulated the first variant of Barequet and Sharir

(1994) as particle filter. Later, they advanced the idea of particle filtering on the space of rigid motions. They use scalar features, that can be calculated streamingly, enabling a streaming pose estimation (Rink et al. 2016).

Papazov et al. (2012) present a 3D object recognition and pose estimation approach for grasping procedures in cluttered and occluded environments. Their method is based on a robust geometric descriptor, a hashing technique and a localized RANSAC-like sampling strategy. Their method simultaneously recognizes multiple model instances and estimates their pose in the scene.

While the pose estimation method itself is not a focus of this work, as the presented approach works with any of them, the ones that were employed here are described in Sect. 3. One of these is a hashing technique as Papazov et al. (2012), while the other is akin to the RANSAC-based geometric consistency checking method employed in Aldoma et al. (2012).

## 2.2 Next-best-view planning

As pointed out by Roy et al. (2004), objects can often not be definitely recognized from one view but require NBV planning (Chen et al. 2011) in order to robustly recognize the object and estimate its pose.

An early active vision system was described by Arbel and Ferrie (2001). They propose an approach for view planning for object classification. It is based on offline generated entropy maps which suggest viewing directions to allow an unambiguous classification of the object at hand. Another approach based on offline computed discriminative views is described by Sipe and Casasent (2002). During a training phase they build so called Feature Space Trajectories which capture how a globally extracted feature changes in relation to changes of the viewing angle. Areas in the feature space where trajectories from different objects come close to each other indicate ambiguity with respect to object classification. Concerning only a single object, areas where trajectories from far apart viewing perspectives are close indicate ambiguity with respect to the object's pose estimate. Inversely, the most discriminative view for disambiguation of two object classes and the most informative view for the pose estimation of an object are extracted offline and used for planning in a similar fashion as in Arbel and Ferrie (2001).

Denzler et al. (2002) describe an object classification system, with a NBV planning approach based on online information theoretic concepts. Instead of precomputing discriminative views offline, the whole approach is centered around an online evaluation of how much a certain camera action and the resulting observation is expected to reduce the current state posterior's uncertainty. This quantity is measured by the mutual information (MI) between the proposed observation and the current state. Although the work of Arbel and Ferrie (2001) and Denzler and Brown (2002) propose

interesting (and in Denzler and Brown 2002 generally optimal) strategies for the selection of camera parameters and viewpoints, the systems have only been verified for low dimensional state spaces or treated pose estimation in an unprincipled way because they were designed for an active classification task. For the practical application of recognizing objects and estimating their poses, view selection and pose estimation go hand in hand as both alter the relative pose between camera and object.

A system explicitly modeling the object's orientation in addition to its class is described by Laporte and Arbel (2006). They also introduce a new view selection criterion which is not based on what effect the measurement has on the state posterior (as in the MI criterion), but on how well it can disambiguate highly probable hypotheses. This idea is similar to the informative views derived by the Feature Space Trajectories in Sipe and Casasent (2002), however here a more probabilistic approach is followed.

The most complete approach described in literature, integrating object classification, 6 DoF continuous pose estimation and NBV planning, to the best of the authors' knowledge, is the work of Eidenberger et al. (2008, 2009, 2012). Formally, the state space is the joint space of  $n$  objects, each with a class label and a 6 DoF pose (modeled as a mixture of 6D multivariate Gaussians). By assuming independence between the objects, the distribution factorizes over  $n$  single object distributions each defined by a discrete distribution over the class label and a class label conditional distribution over the pose space. The measurement model allows for prediction of feature locations in the image plane given constellations of multiple objects and a camera viewpoint. This way and by building the measurement model not in 6 DoF pose space but in the image coordinate space of the features, occlusions between the objects in the scene can be taken into account. NBV planning is addressed as decision making problem for an agent in a partially observable environment which needs to select a new observation pose (the planning space is heavily discretized using only in the order of 10 possible camera poses). This leads to the formal framework of a partially observable Markov decision process (POMDP), where the expected future reward is maximized by following an optimal action policy which maps the current state belief to an action. Pose and object detection takes 0.9 seconds, and the consequent use of parametric distributions, a closed form upper bound approximation for the costly information gain summand of the reward function and the number of real experiments suggests a planning time in the order of seconds.

The proposed histogram-based viewpoint selection approach, however, can be pre-computed and affords a constant-time lookup in each step (based on the current most likely estimate). This property makes it very attractive for mobile robotic applications, where computing power and time might be limited.

### 2.3 Integration of multiple estimates

Romea et al. (2010) present an approach for recognizing all objects in a scene and estimating their full pose from multiple views. Their approach builds upon a single-view algorithm which recognizes and registers learned metric 3D models using local descriptors. The work is extended to multiple views using a multi-step optimization that processes each view individually and feeds consistent hypotheses back to the algorithm for global refinement. However, the multiview approach is based on a fixed camera setup. Similarly, Selinger and Nelson (2001) try to improve object recognition results in a multiple fixed camera setup. They apply several constraints to the object and the scene and choose the object hypothesis with the highest confidence.

Voit et al. (2006) present a system for estimating human head pose with the use of multiple camera views. Thereby, a neural network is applied to each of the views to fuse the output using a Bayesian filter framework. While the approach of using a Bayesian filter to improve the accuracy of the human head pose is a similar idea to our approach, the realization is much more limited. Their state space only consists of 360 individual horizontal head orientations.

Vikstén et al. (2006) present a system which integrates the information output from several appearance-based pose estimation algorithms and from several views of the scene. The computation of a final orientation by considering the orientations from several views is done by computing the average of the last 5 successive detections. Thus, it is basically the same as the approach in our previous work (Kriegel et al. 2013), where all detections are clustered and the largest averaged.

Some pure pose estimation techniques are also strongly related to object detection and our approach of fusing pose estimations from different views. For example, Glover et al. (2011) model the space of orientations based on multiple evidences, but in their case these come from local features in a single dataset. Also, they model the space of quaternions using Bingham distributions (intuitively a Gaussian-like distribution on a hypersphere).

Generally, all Bayesian filters can be updated by evidences from multiple views. However, in Sect. 7.2 we evaluate a particle filter, pose clustering and averaging, and two Bayesian variants of a multi-view orientation estimation based on Bingham distributions, and the results suggest that the proposed histogram filter based approach is more accurate than the others.

### 3 Employed pose estimation approaches

The histogram filter, which is described in Sect. 5, uses detected orientations  $R \in SO(3)$  of the object to update the hypothesis of its actual orientation. These orientation estimates come from a pose estimation method. While the

presented work is independent of the used pose estimator, we briefly present the two that were used during the evaluation in Sect. 7. These assume that the object of interest has already been segmented from the background and is represented as a set of 3D points (typically a cluster of points).

#### 3.1 Geometric hashing

A generic method operating on dense depth images was implemented (Kriegel et al. 2013). The meshes of objects of interest can be either retrieved from CAD data, or generated during autonomous object modeling (see Kriegel et al. 2013).

The developed geometry based object recognition method is based on the work of Drost et al. (2010). A global model for each object is built using a feature similar to the surflet pair features (Wahl et al. 2003). Specifically, the feature is a four dimensional vector  $\mathbf{F}$  representing the geometrical relation between the distance vector  $d = m_2 - m_1$  between two points  $(m_1, m_2)$  and their normals  $(n_1, n_2)$ :

$$\mathbf{F}(m_1, n_1, m_2, n_2) = (|d|, \angle(n_1, d), \angle(n_2, d), \angle(n_1, n_2)) \quad (1)$$

where  $\angle(a, b)$  denotes the angle between two vectors.

For each of the objects to be detected, a model is generated by randomly drawing a subset of the object's surface points and calculating  $\mathbf{F}$  for all possible combinations of point pairs in the subset. The generated features are discretized and used as a key in a four dimensional hash table storing the point pairs.

During object detection, a random reference point  $s_r$  is chosen in the examined 3D point cluster and paired with all other points  $s_i$  in a randomly sampled subset of the points in the cluster. For each of these point pairs, similar pairs  $(m_r, m_i)$  are retrieved from the precomputed model hash tables. Presuming a correspondence between  $s_r$  and  $m_r$ , and taking their respective normals  $n_{s_r}$  and  $n_{m_r}$  into consideration, the pose of the object in the scene is defined up to a rotation around  $n_{s_r}$ . The magnitude of this rotation is calculated by aligning  $s_i$  and  $m_i$ . Subsequently, for each matched pair of point pairs a vote for a correspondence between  $s_r$  and  $m_r$  as well as the calculated angle is cast in a two dimensional space. The most dominant peaks are considered to be hypotheses for the poses of an object.

Since there is a chance that the selected  $s_r$  is unrepresentative (e.g. a point on large plane), the process is repeated several times. Finally, quality values for all generated hypotheses are calculated by rendering the objects in their estimated poses and pixel-wise comparing the resulting depth buffer with the acquired depth data. This score is computed efficiently on the GPU, and if it surpasses a threshold, the highest ranked hypothesis is accepted as a fit.



### 3.2 Correspondence filtering using RANSAC

In Rusu et al. (2009), a Sample Consensus based method for the initial alignment (SAC-IA) of two datasets is presented. The SAC-IA method tries to maintain the same geometric relations of the correspondences as defined in Rusu et al. (2008) without having to try all combinations of a limited set of correspondences. Instead, large numbers of correspondence candidates (corresponding point triplets) are sampled and each of them are ranked:

1. Select  $s = 3$  sample points from  $\mathbf{P}$  while making sure that their pairwise distances are greater than a user-defined minimum distance  $d_{\min}$ .
2. For each of the sample points, find a list of points in  $\mathbf{Q}$  whose feature descriptors are similar to the sample points' descriptors. From these, select one randomly which will be considered that sample points' correspondence.
3. Compute the rigid transformation defined by the sample points and their correspondences and compute an error metric for the point cloud that computes the quality of the transformation.

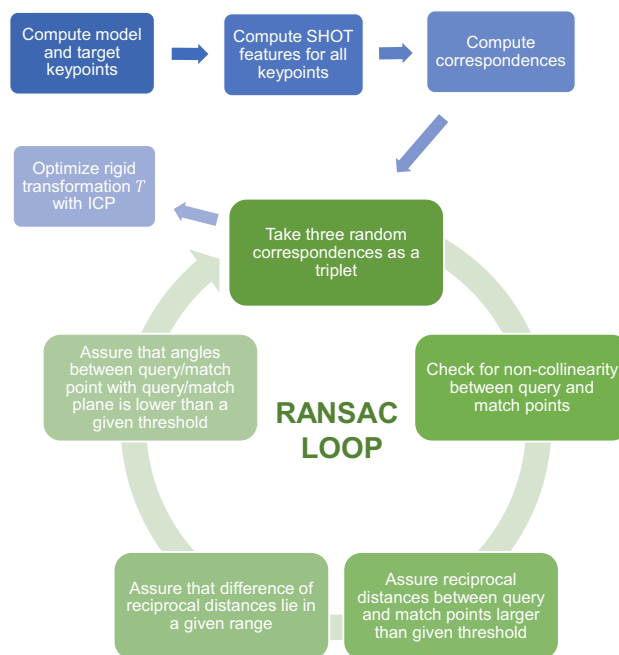
These three steps are repeated, and the transformation that yielded the best error metric is stored and used to roughly align the partial views. Finally, a non-linear local optimization is applied using a Levenberg–Marquardt algorithm (Fitzgibbon 2001).

For this work, a similar approach has been implemented, consisting of the following steps:

1. Compute model keypoints  $\mathbf{p}_{k,m}$  and target keypoints  $\mathbf{p}_{k,t}$  (e.g. simply subsampling the point cluster).
2. Compute the feature descriptors for each keypoint (we used the SHOT feature (Tombari et al. 2010), following the evaluation in Aldoma et al. 2012).
3. Compute  $k$  correspondences between model keypoints  $\mathbf{p}_{k,m}$  and target keypoints  $\mathbf{p}_{k,t}$  and remove duplicate correspondences.
4. Use RANSAC to optimize the rigid transformation  $T$  between the sample points and their correspondences. The randomized selection of the correspondences is guided via several conditions these have to fulfill. The RANSAC optimization stops after it is unlikely that a more optimal  $T$  than the current one can be found.

The last step (and the conditions which the correspondences have to fulfill) consists of:

1. Select three random correspondences as a triplet.
2. Check for non-collinearity between the query points  $\mathbf{p}_q$  and the match points  $\mathbf{p}_m$  in the triplets.

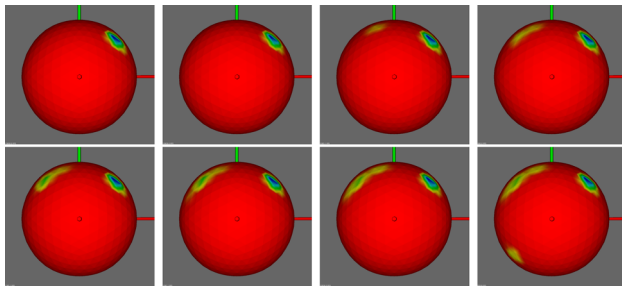


**Fig. 2** Overview of the local correspondence and RANSAC based approach. First, keypoints, features and correspondences are computed, which afterwards are used in the RANSAC-loop to compute the most likely rigid transformation  $T$  between the model and the target. At the end, ICP is performed to optimize  $T$

3. Assure that the reciprocal distances between the query points and the match points are larger than a given threshold:  $d_{q_i,j} = \|\mathbf{p}_{q_i} - \mathbf{p}_{q_j}\| \geq t_r$  and  $d_{m_i,j} = \|\mathbf{p}_{m_i} - \mathbf{p}_{m_j}\| \geq t_r$ .
4. Assure that the difference of  $d_{q_i,j}$  and  $d_{m_i,j}$  is small:  $|d_{q_i,j} - d_{m_i,j}| \leq t_d$
5. Assure that the differences of corresponding angles between the query points and match points are smaller than  $t_\alpha$ .

These conditions ensure that well selected correspondences are considered for estimating the rigid transformation  $T$ . After the RANSAC-loop found the maximal consistent subset of correspondence, the final transformation is computed. As with global registration algorithms in general, the Iterative Closest Point (ICP) algorithm (Chen and Medioni 1992) can be applied in order to further optimize the transformation between the model and the target. In Fig. 2 an overview for this approach is illustrated.

An important advantage of this algorithm is the capability of adjusting the runtime easily for one pose estimation. The number of iterations of the RANSAC-loop depends on the desired probability of success. Setting this value close to 1 (i.e. above 99.9%) results in good estimates at the cost of increased runtime. Low values increase the chance of suboptimal pose estimates, but at markedly shorter runtimes. While this method is less accurate than the geometric hashing one, it



**Fig. 3** Example for visualizing the weights  $w_r$  of the representative orientations  $\mathbf{q}_r$  for multiple detections  $\mathbf{q}_d$ , showing the accumulation of weights in each step (*red* signals 0, while *blue* the highest density regions) (Color figure online)

was chosen for performing the large number of trials needed for the simulated experiments in Sect. 7.3.

## 4 Representation and sampling of rotations

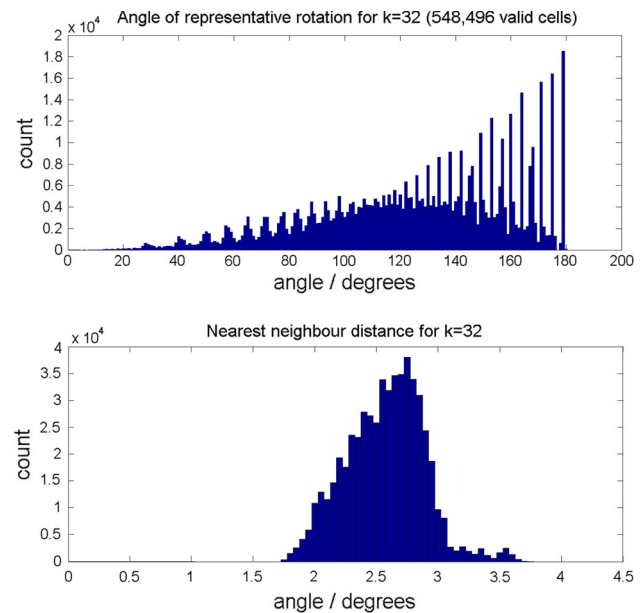
The discrete Bayes filter used in this work computes an orientation  $R \in \mathcal{R}$  for multiple detected orientations  $R_d \in SO(3)$ . This filter behaves almost the same as a *particle filter* but for the histogram filter the particles are fixed, which has the advantage that the particles do not deplete. The histogram filter only estimates the orientation of the object and not the position. Since state-of-the-art pose estimators can estimate the real position very accurately, there is a greater need for an orientation estimator rather than a position estimator. We represented these orientations using quaternions (4D unit vectors  $\mathbf{q} = [x, y, z, w]$ ) due to their benefits, e.g. simple (approximate) weighted averaging (Sharf et al. 2010) (see Eq. 7).

### 4.1 Visualization of rotations

For visualizing distributions over quaternions, the *visualization module*<sup>1</sup> of the *Point Cloud Library* (PCL) was used. Since visualizing orientations  $\mathbf{q} \in SO(3)$  compactly is difficult, the orientations to be represented are applied to a unit vector  $\mathbf{v}_1 = [1, 1, 1]/\sqrt{3}$  and the resulting vectors' density is color-coded on the unit sphere.

In Fig. 3 this visualization of the orientations from a series of detections are shown. In the first two upper left figures the orientations are around identity (the ground truth orientation). In the next five figures the orientations rotated 90° around the  $z$ -axis gain more and more weight. In the last figure a detection rotated 180° around the  $z$ -axis was added. Looking at these figures, one gains some insight about the detected orientations, even though all rotations around  $\mathbf{v}_1$  are mapped to the same spot. Intuitively, the shown detections

<sup>1</sup> [http://docs.pointclouds.org/1.7.0/group\\_\\_visualization.html](http://docs.pointclouds.org/1.7.0/group__visualization.html).



**Fig. 4** Statistics for the grid-based sampling approach for  $k = 32$  and therefore 548,496 valid orientations. The angles of the representative orientations  $\mathbf{q}_r$  (*top*) and the nearest neighbor distances are shown (*bottom*)

suggest that the used object has probably similar-looking poses at 0°, 90° and 180° rotations around the  $z$ -axis.

### 4.2 Histogram of rotations

A deterministic, and preferably uniform sampling of the  $SO(3)$  space is needed in order to use the histogram filter. To represent the orientations, we evenly divided the space of quaternions, and selected those cells that contain unit quaternions. This does not result in a perfectly even sampling, but the area of  $SO(3)$  that falls into each cell was estimated, and used as a uniform prior.<sup>2</sup> We divided the ranges of the 4D quaternion dimensions, i.e.  $\pm 1$  and considering only half of it for  $w$ , into  $2k$  and  $k$  divisions, respectively, resulting in a grid of size  $2k \times 2k \times 2k \times k$ .

In order to visualize the distributions of the representative orientations  $\mathbf{q}_r$ , the angle  $\theta$ , describing the magnitude of the orientation (w.r.t identity), and the distance  $d_{nn}$  from an orientation to its nearest neighbor are used. In Fig. 4 the histograms of  $\theta$  and  $d_{nn}$  for  $k = 32$  are shown. For  $\theta$  the discretization steps are evident, as the distributions should be  $\sin^2(\theta/2)$  (area of a sphere cap), a function that is almost linear in the middle region. Though, this discretization does not affect the effectiveness of the histogram filter too heavily,

<sup>2</sup> In our experiments the prior was always uniformly distributed, but it can be easily modified to any distribution the discretization allows to represent (which is much more flexible than a parametric model).

**Table 1** Statistics of nearest neighbor distances  $d_{nn}$  (the mean  $\mu$  and the standard deviation  $\sigma$ ), and the memory usage (RAM) for different  $k$ 's are shown ( $N$  is the corresponding number of orientations used)

$k$	$N$	$\mu$ ( $^\circ$ )	$\sigma$ ( $^\circ$ )	RAM (MB)
8	8480	10.28	1.40	~48
16	68,416	5.12	0.65	~68
32	548,496	2.57	0.32	~270

since the  $\mathbf{q}_r$ 's are averaged to get a final orientation from the histogram filter, as described in Sect. 5.2.

Instead, the nearest neighbor distances  $d_{nn}$  (i.e. the density of the  $N$  cell centers) are more critical to the performance. In the second plot in Fig. 4, the  $d_{nn}$  histogram shows the (not perfectly peaked) resolution of the sampling approach. In Table 1 the mean and the standard deviation for  $k = 8$ ,  $k = 16$  and  $k = 32$  are shown. When doubling the factor  $k$  (halving the step length  $l_s$ ), the mean  $\mu$  and the standard deviation  $\sigma$  is halved. Additionally, the relative increase in  $k$  is producing roughly cubically larger relative increase in  $N$  (as it influences the sampling of the 4D half-hypersphere's surface, which is a 3D space).

A desirable resolution would be if the mean was about  $\mu_{des} \approx 1^\circ$ . So,  $\mu_{k=32} = 2.57^\circ$  is not sufficient and  $\mu_{k=64}$  should be considered. The mean for  $k = 64$  would only be  $\mu_{k=64} \approx 0.5 \cdot \mu_{k=32} \approx 1.3^\circ$ . But in Table 1 it is shown that the memory usage for  $k = 32$  is already 270 MB, and for  $k = 64$  it was already over 2 GB, which is not a reasonable trade-off (note that the full grid is saved for optimizing access time, resulting in some excessive memory usage, roughly proportional to  $N^{4/3}$ ).

## 5 Histogram filter based orientation fusion

Histogram filters are related to particle filters, but work on the cells of the discretized search space, which can be interpreted as a fixed set of particles. Thus, they cover the whole search space and avoid the problem of particle depletion, but come at the cost of a fixed number of particles, resulting in two major drawbacks. On the one hand a huge number of them may be needed. On the other hand the discretization limits the accuracy. In the case of pose estimation, the search-space is the 6 degrees of freedom (DoF) pose space. As this is quite large, several pose estimation methods separate the position and orientation parts (see related work).

Here we abstract away from the pose estimation methods, and treat only their output as estimates of the pose, and integrate the orientation estimates using a histogram filter. As mentioned earlier, the histogram filter only treats the orientation part of the 6 DoF pose, which is estimated with a pose estimator since the position can be processed with

common techniques, such as clustering and averaging, as presented in Kriegel et al. (2013). The disadvantage is that we are neglecting possible correlations between the position and orientation, i.e. position-dependent orientation errors and orientation-dependent position errors. The former could arise for example due to grossly deteriorating depth measurements at great distances, and the latter from object centroids being estimated incorrectly for large rotational errors. Neither of them is overly critical, as pose estimation is typically done at close-range, and a combination of a highly accurate position and orientation yields a good pose.

The proposed method considers the measurement (and movement<sup>3</sup>) model in the updating step, when it fuses detections from multiple viewpoints. As errors from different views have a large chance of being uncorrelated, their combination increases the overall accuracy, as we have shown in Kriegel et al. (2013).

### 5.1 Histogram filter reweighting

After a new orientation  $\mathbf{q}_d$  is detected, the cell weights  $w_r$  of the histogram filter are updated. The update is done with a truncated Gaussian measurement/error model. The confidence weight (CW)  $w_c$  is computed by

$$w_c = f_{w_c}(d_q | \sigma_{cw}) = e^{-\frac{d_q^2}{2\sigma_{cw}^2}}, \quad (2)$$

with  $d_q$  being the distance between two orientations and  $\sigma_{cw}$  the standard deviation for the confidence weight. The pre-factor of the Gaussian  $a_G = 1/\sigma\sqrt{2\pi}$  is left out in the calculation, causing the function to start at 1.0. Since  $d_q \leq 180^\circ$ , the lower limit of  $w_c$  depends on  $\sigma_{cw}$ .

According to the Bayes update rule, updating a weight with a new detection is done by a multiplication with  $w_c$ . However, this results in extremely low weights if detections do not follow the assumed Gaussian error distribution, which is very often the case (see discussion in Sect. 6). Therefore, a different, non-Bayesian fusion method is employed, inspired by ensemble learning fusion rules, which proved to be more robust than probability multiplications (at least when these probabilities were only approximately correct) (Marton et al. 2012). Instead of multiplying the step weights, they are summed up in each step, representing that either of them could be a correct or incorrect one. Thus the weight  $w_r^T$  at current step  $T$  is the sum of all normalized weights  $\hat{w}_c^t$ :

$$w_r^T = \sum_{t=0}^T \hat{w}_c^t \quad (3)$$

<sup>3</sup> We do not apply an additional uncertainty to the estimates after a viewpoint change, as we were using very precise sensor movements to obtain ground truth.

Additionally, a *verification weight* (VW)  $w_v$  can be multiplied to the cell weights  $w_r$ . This weight indicates the correctness of the detection. The pose estimator introduced in Sect. 3 provides a measurement, which exactly describes the correctness of the detection. It is the mean squared error (MSE) of all points from the model to the aligned target. The verification weight  $w_v$  then looks as follow:

$$f_{w_v}(\text{MSE}, \sigma_{vw}) = w_v = e^{-\frac{\text{MSE}^2}{2\sigma_{vw}^2}}, \tag{4}$$

with  $\sigma_{vw}$  being the standard deviation for the verification weight. As in Eq. (2) only a Gaussian-like error model is used because weights are desired, where the maximum is 1.0. Using Eq. (4) the computation for each cell weight in Eq. (3) can be extended to:

$$w_r^T = \sum_{t=0}^T \hat{w}_c^t \cdot w_v^t \tag{5}$$

### 5.2 Final rotation computation

In order to finish the detection process, a termination condition has to be declared. As a general rule, the error  $\epsilon_f = d(\mathbf{q}_{f_p}, \mathbf{q}_{f_c})$  from the previous final orientation  $\mathbf{q}_{f_p}$  to the current final orientation  $\mathbf{q}_{f_c}$  is computed and based on that it is decided whether to continue or terminate the process. But before the error can be estimated, the final orientations  $\mathbf{q}_f$  has to be computed. There are multiple possibilities to compute  $\mathbf{q}_f$ :

1. voting
2. verification weighted voting
3. confidence weighted clustering and averaging
4. confidence and verification weighted clustering and averaging

For the voting-based computation of  $\mathbf{q}_f$ , in each step the detected orientation  $\mathbf{q}_d$  is discretized into its corresponding cell. The weight for this cell is then incremented with 1.0:  $w_r = w_r + 1.0$ . The same operation is done when using verification weighted voting. But instead of incrementing the weight with 1.0, it is incremented with the corresponding verification weight  $w_v$  computed with Eq. (4):  $w_r = w_r + w_v$ . Both, voting and verification weighted voting, have one common deficit: discretization errors of the sampling approach affect the computation of  $\mathbf{q}_f$ .

In order to overcome the discretization errors for  $\mathbf{q}_f$ , confidence weighted clustering and confidence and verification weighted clustering is deployed. This is achieved with a simple region growing algorithm:

1. search for cells  $\mathbf{c}_{max}$  with a weight  $w_r \geq w_{max} \cdot w_M$ , where  $w_{max}$  is a predefined proportion of the maximal weight  $w_M \implies$  add the corresponding weighted representative orientation  $\mathbf{q}_{r,w}$  to a new cluster  $\mathbf{C}$
2. search for neighboring cells  $\mathbf{c}_{max_{nn}}$  with a weight  $w_r \geq w_{min} \cdot w_M$ , where  $w_{min}$  is a predefined proportion threshold for checking if a cell is still belonging to the seed cluster  $\implies$  add those  $\mathbf{q}_{r,w}$ 's to the cluster  $\mathbf{C}$
3. add all clusters  $\mathbf{C}$  to the list of possible clusters  $L_C$

The weighted representative orientation  $\mathbf{q}_{r,w}$  is the product of  $\mathbf{q}_r$ , its corresponding cell weight  $w_r$  and prior cell weight  $w_I$  (with  $w_I$  being proportional to the cell area in the case of a uniform prior):

$$\mathbf{q}_{r,w} = \mathbf{q}_r \cdot w_r \cdot w_I. \tag{6}$$

The final orientation  $\mathbf{q}_f$  can then be estimated as the average of the most likely cluster  $\mathbf{C}_{max} \in L_C$ :

$$\mathbf{q}_f = \sum_{i=0}^N \mathbf{q}_{r,w_i}^{C_{max}} / \left\| \sum_{i=0}^N \mathbf{q}_{r,w_i}^{C_{max}} \right\|, \tag{7}$$

computed using the accurate closed-form approximation from Sharf et al. (2010), with  $\mathbf{q}_{r,w_i}^{C_{max}}$  being the  $\mathbf{q}_{r,w}$  for  $\mathbf{C}_{max}$ .

## 6 Measurement model, symmetry estimation and viewpoint selection

In the previous section, a Gaussian-like measurement error model was described. However, this is only an approximation to the real error model, since many man-made objects have symmetrical structures. In order to estimate the real error model, those symmetrical structures of the object has to be incorporated.

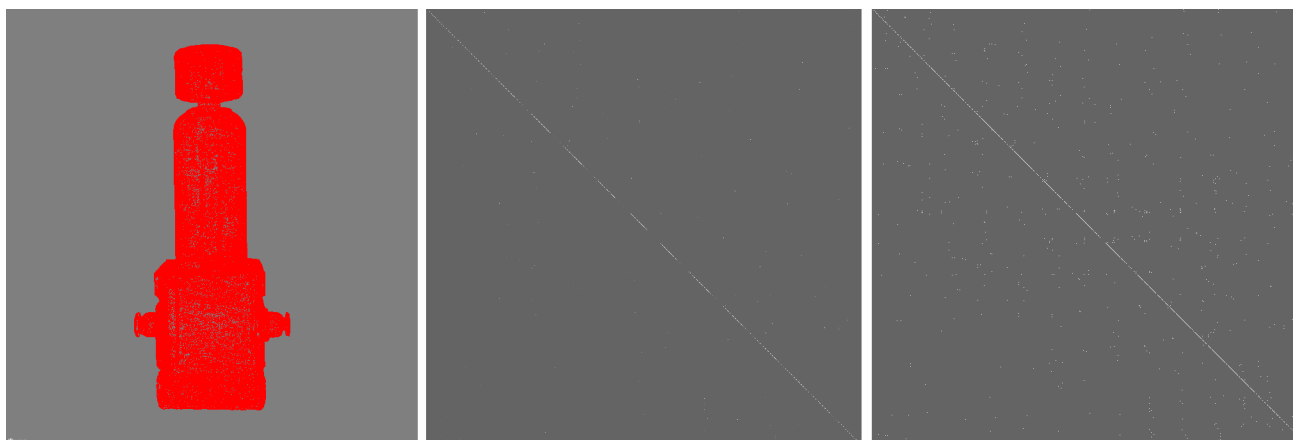
With objects having symmetrical structures, pose estimators tend to have three sources of errors:

1. random errors
2. pose estimator's systematic errors
3. errors due symmetrical structures

The random errors can be approximated by a Gaussian error model, which has been covered in the previous section. The other two error sources can be solved by a correct estimation of the error model, by computing the confusions between the representative orientations  $\mathbf{q}_r$ . These confusions can be expressed with a *contingency table* (CT), which is a multi-valued confusion matrix.

As for representing a multi-label classification performance, a contingency table is used to describe the confusion





**Fig. 5** Example contingency table for the object *filter* (left). In the *middle* the complete CT is shown and on the *right* the CT is shown, where all non-zero values are set to the maximum

between two representative orientations  $\mathbf{q}_{r,i}$  and  $\mathbf{q}_{r,j}$ . The contingency table is structured as follows:

- rows: ground truth orientations  $\mathbf{q}_{GT} \in \mathcal{R}$
- columns: detected orientations  $\mathbf{q}_D \in \mathcal{R}$
- entries: probability  $P_{i,k}$ , how often  $\mathbf{q}_{Dk}$  was detected when  $\mathbf{q}_{GT_i}$  was the real orientation

Therefore, the contingency table is a quadratic matrix with  $CT \in \mathbb{N}^{N \times N}$ , with  $N = |\mathcal{R}|$ ,  $N \in \mathbb{N}$  being the number of representative orientations.

In Fig. 5 a contingency table for an example object and for  $k = 8$  (8480 orientations) is shown. The object has some symmetrical structures:

- the identity orientation looks the same as the one rotated  $180^\circ$  around the  $z$ -axis;
- the  $90^\circ$  rotated orientation around the  $z$ -axis looks the same as the one rotated  $270^\circ$  around the  $z$ -axis.

Most of the non-zero entries appear on the diagonal of the contingency table, meaning that for most of the orientations there are just a few confusions.

### 6.1 Histogram filter reweighting

In order to reweight the histogram filter according to the probabilities from the contingency table, Eq. (2) has to be reformulated.

When the orientation  $\mathbf{q}_D$  with the viewpoint  $\mathbf{q}_v$  was detected, the probabilities for the ground truth orientations are approximated by:

$$P(\mathbf{q}'_{GT_i} | \mathbf{q}'_D) \approx CT(\mathbf{q}'_{GT_i}, \mathbf{q}'_D). \tag{8}$$

In other words, the probabilities of the detected column are taken to reweight the cells of the histogram filter with the help of the contingency table.

Thus the  $i$ -th accuracy weight (AW)  $w_{a,i}$  is:

$$w_{a,i} = f_{w_{a,i}}(\mathbf{q}'_D) = CT(\mathbf{q}'_{GT_i}, \mathbf{q}'_D) + c_p. \tag{9}$$

Since the entries of the contingency table are empirical determined values, the correctness of these values are not fulfilled completely. Thus, a pseudo-count  $c_p$  is added to the accuracy weight  $w_a$ . Using Eq. (9) the  $i$ -th weight of the cells at time step  $T$  can be formulated as:

$$w_{a,i}^T = \sum_{t=0}^T \hat{w}_{a,i}^t. \tag{10}$$

### 6.2 Contingency table estimation

The entries of the contingency table are estimated with computing the confusions between the representative orientations  $\mathbf{q}_r$ . There are two possible techniques to compute the confusions between two orientations. Either the scans of the object, for which the contingency table is going to be computed, are acquired. The alternative would be to simulate the scans of the objects, for example with the *simulation module*<sup>4</sup> of PCL. This way, the individual scans can be generated very easily. The generation of real world scans with a scanning device would be very tedious, thus the sensor simulation is used to acquire the individual scans of the object. The simulation module of PCL simulates a “Xtion-like” scan for a given camera viewpoint  $\mathbf{v}_p$ .

<sup>4</sup> <https://github.com/PointCloudLibrary/pcl/tree/master/simulation>.

For the estimation of the contingency table the object has to be scanned several times for each orientation  $\mathbf{q}_r$ . Therefore, a minimum number of scans for each orientation has to be computed beforehand, in order to guarantee a narrow confidence interval for the cells holding no detections.

The tests can be treated as Bernoulli trials for each cell separately, representing if the given orientation was detected or not. Then, a series of tests have a binomial distribution, with parameter  $p_b$  indicating the probability of that orientation being detected. The conjugate prior of the binomial distribution is the Beta distribution, which is modeling the distribution over  $p_b$ . This means that a Jeffreys interval can be computed for  $p_b$  given a significance level, which in this case this is equivalent to the Bayesian credible interval. The non-informative Jeffreys prior is  $\text{Beta}(0.5, 0.5)$ , and it gets updated using Bayes theorem after each trial. In the case of the Beta distribution this fortunately yields another Beta distribution:  $\text{Beta}(0.5, n + 0.5)$ , where  $n$  is the number of steps (trials). Thus, the minimum number of trials can be found, s.t. the confidence/credible interval for the 0 value is in the interval  $[0, 5\%]$ :

$$\min \left\{ n | n \in \mathbb{N}, \int_0^{0.05} \text{Beta}(0.5, n + 0.5) \geq 1 - \alpha \right\}. \quad (11)$$

Since multiple cells are tested for confusions, Bonferroni correction has to be performed as well.

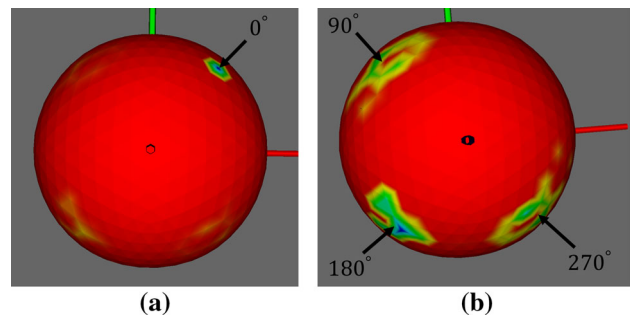
For example, for 8480 orientations ( $k = 8$ ) and a significance level of  $\alpha = 5\%$ , this results in  $n = 250$  trials and allows for mistakes of up to 6 missed hits (for 8480 possible detections) in 95% of the cases for the cells that were not hit during evaluation. Thus  $c_p$  can be chosen to be around 6, representing the uncertainty we have when lacking observations. For  $k = 32$  there are 548,496 valid orientations, requiring a minimum number of  $n = 300$  trials each. Since this would be too time-consuming, we estimated a measurement model only with  $k = 8$ , i.e. for orientations at roughly every  $10^\circ$  (see Table 1).

### 6.3 Symmetry estimation

The contingency table can also be used to estimate the symmetries of the object. The symmetries for the objects are expressed by a weights vector  $W_s$ , where each entry is a corresponding weight for the representative orientations  $\mathbf{q}_r$ . These weights indicate the difference  $\mathbf{q}'$  rotation, which was the case for a confusion between two orientations  $\mathbf{q}_{GT_i}$  and  $\mathbf{q}_{D_k}$ :

$$\mathbf{q}' = \mathbf{q}_{GT_i}^* \cdot \mathbf{q}_{D_k}. \quad (12)$$

After discretizing the difference rotation  $\mathbf{q}'$ , the corresponding weight is added with the corresponding entry in the contingency table (Fig. 6):



**Fig. 6** The symmetry weights for the object *filter* are shown in both figures. In (a) all symmetry weights are shown. Only one noteworthy peak is visible, which is at identity, meaning that for most of the representative orientations  $\mathbf{q}_r$  no confusions are detected. In (b) the weights around identity are left out, such that the other weights are better visible. Three other peaks are now visible, at  $90^\circ$ ,  $180^\circ$  and  $270^\circ$  around the  $z$ -axis

$$W_s(\mathbf{q}') += \text{CT}(\mathbf{q}_{GT_i}, \mathbf{q}_{D_k}). \quad (13)$$

The symmetries of an object can be estimated in a different way as well. Again, the weights vector  $W_s$  describes the symmetries of the object. Now,  $W_s$  is computed by rotating the object with each of the representative orientations  $\mathbf{q}_r$  and afterwards computing the point-wise mean squared error of the initial model  $O_I$  and the rotated model  $O_r$ . The weight  $w_s \in W_s$  is again the corresponding weight to the difference rotation  $\mathbf{q}'$ , which is in this case simply the representative orientation  $\mathbf{q}_r$ . It can be computed by a Gaussian-like function with the mean squared error as input:

$$w_s = e^{-\frac{\text{MSE}(O_I, O_r)^2}{2\sigma_{\text{mse}}^2}}, \quad (14)$$

with  $\sigma_{\text{mse}}$  being the standard deviation for the mean squared error. The drawback of this method is that only the symmetries of the objects are estimated, while when estimating the symmetries with the contingency table, the pose estimators systematic error's are included as well. However, the mean squared error based estimation can be done much faster, since the computation of the contingency table is rather time consuming.

After estimating the symmetries of an object, one can further process the symmetry weights or use the symmetry weights directly. For example, a possible approach to compute the symmetry axis could be to run a principal component analysis (PCA) or fitting Bingham distribution(s) to the symmetry weights  $w_s$  (or in this case, on the representative orientations  $\mathbf{q}_r$ ).

### 6.4 Next-best-view estimation

The contingency table of the measurement model can also be used in an algorithm that estimates a next-best-view (NBV)

with a lowest confusion between the detected orientation  $\mathbf{q}_D$  and the ground truth orientations  $\mathbf{q}_{GT}$ . This increases the chance that each detection used by the histogram filter improves the final orientation  $\mathbf{q}_f$ .

In each step, the algorithm tries to verify that the most likely object pose  $\mathbf{q}_f$  is the correct one. Therefore it is checking in the contingency table where should the sensor be placed in order to observe the object from a viewpoint that is the least ambiguous (i.e. shows the highest precision and recall). This is achieved by looking at the distribution over detected orientation given each ground truth pose, and marginalize the likelihood of an ambiguous detection over the possible detections. The likelihood of an ambiguous detection is then simply taken as the number of times the given detection was reported incorrectly when building the contingency table. In the ideal case, this heuristics selects the orientation with the least false positives and false negatives.

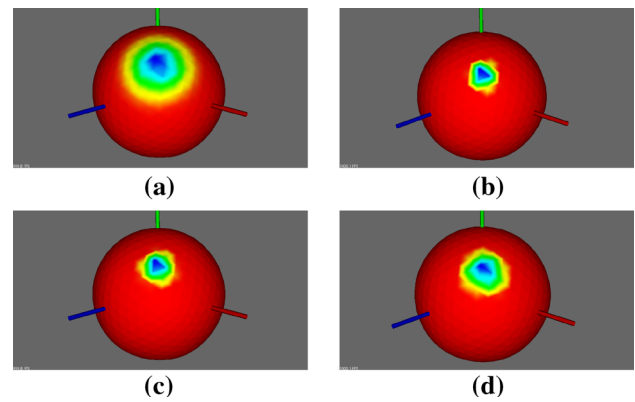
Since the problem is simplified by not considering the current distribution over possible orientation and focusing only on the most likely one, the ordering of the viewpoints can be precomputed as well. This way the NBV selection is a very efficient lookup, and the algorithm is rotating the current viewpoint s.t. to observe the selected orientation of the object (assuming that  $\mathbf{q}_f$  is the correct current viewpoint). After the detection is made,  $\mathbf{q}_f$  is updated, and a new lookup and move is performed (avoiding to place the sensor repeatedly in the same orientation).

## 7 Evaluations and comparisons

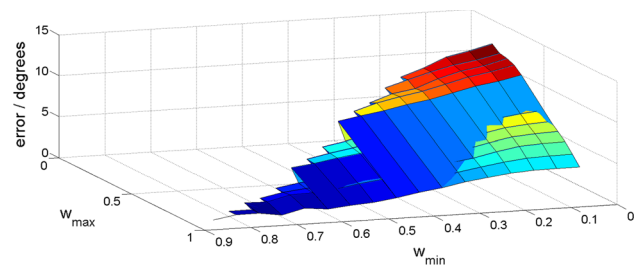
In this section, the introduced histogram filter variants are evaluated and compared to alternatives, based on several real and simulated sensors. Before reporting the results, the parameters for detecting the largest peak are discussed in the next subsection. Then, real world results and comparisons to alternative methods are shown in Sect. 7.2, followed by a large-scale evaluation of the algorithm variants in Sect. 7.3. Section 7.4 presents the evaluation of the contingency table measurement model, while Sect. 7.5 that of the next-best-view selection.

### 7.1 Clustering parameters

When using different values for the parameters  $w_{min}$  and  $w_{max}$ , the size of the most likely cluster  $\mathbf{C}_{max}$  differs and therefore, a different final orientation  $\mathbf{q}_f$  is obtained. In Fig. 7 a visualization of the maximum cluster for different parameters is shown. In Fig. 7a, all weights at the end of the reweighting phase are shown. In Fig. 7b–d, the weights of the most likely cluster are shown when using the specific parameters.



**Fig. 7** Visualization of different parameters for the clustering's effect on the detected maxima of the histogram. **a** Initial histogram filter weights after the final step. **b** Weights of  $\mathbf{C}_{max}$  for  $w_{min} = 0.8$  and  $w_{max} = 0.9$ . **c** Weights of  $\mathbf{C}_{max}$  for  $w_{min} = 0.6$  and  $w_{max} = 0.9$ . **d** Weights of  $\mathbf{C}_{max}$  for  $w_{min} = 0.4$  and  $w_{max} = 0.9$



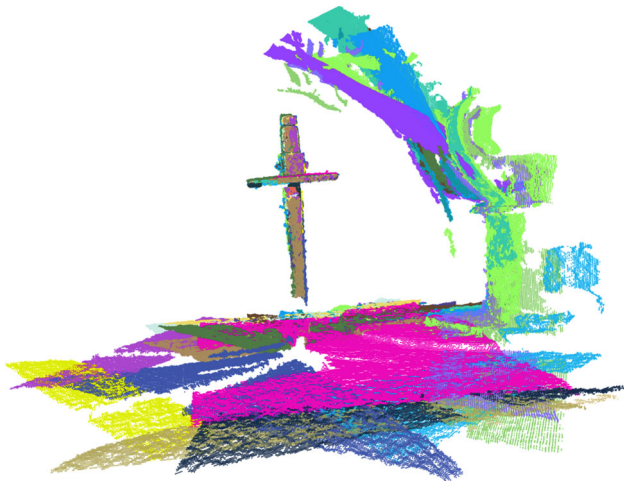
**Fig. 8** Evaluation for different  $w_{min}$  and  $w_{max}$  combinations. The rotational errors are the mean of all the errors for the objects in Fig. 12

An initial evaluation was made to determine the best parameters, which leads to the best most likely cluster  $\mathbf{C}_{max}$ , hence, to the best final orientation  $\mathbf{q}_f$ . In Fig. 8 the mean errors of the final orientations for all evaluated objects (see Fig. 12) are shown, when using the different parameters  $w_{min}$  and  $w_{max}$ . The higher both  $w_{min}$  and  $w_{max}$  get, the better the rotational error gets. For  $w_{min} = 0.1$  and  $w_{max} = 0.2$  the mean rotational error is approximately  $\Delta e(\mathbf{q}_f) = 12^\circ$ . The best results have been achieved when using  $w_{min} = 0.7$  and  $w_{max} = 0.9$ , where the mean of the rotational errors is  $\Delta e(\mathbf{q}_f) \leq 0.1^\circ$ . The evaluation with these parameters will be shown in the following.

### 7.2 Real-world experiments and alternative methods

To test the presented ideas, several Asus Xtion frames were captured using an industrial robot (see Fig. 9), where ground truth object poses were known and the changes in camera positions could be accurately measured (thus the motion uncertainty was negligible). More information on the setup can be found in Kriegel et al. (2013).

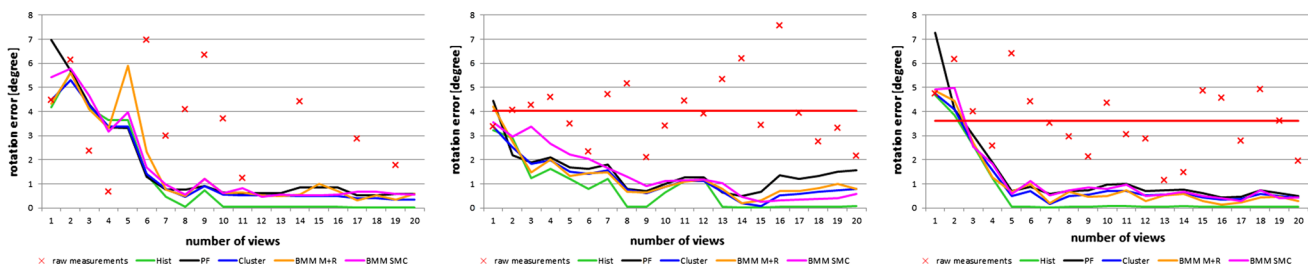
In Fig. 10, the evaluation of the three industrial objects, *filter*, *pressure* and *control*, from Fig. 1 are shown. The rotation



**Fig. 9** The orientation estimation was evaluated in depth images recorded by the industrial arm (visible on the *right* in some of the scans). Here the scans of the *filter* object from Fig. 1 are shown

errors for each detection are marked with a red star. The red line indicates the mean of the detection errors. The clustering method is the nearest-neighbor clustering approach from Kriegel et al. (2013), where the detections are clustered in a nearest-neighbor-manner and at the end the average for the cluster with the most detections is computed. These errors are marked by the blue line. The green line indicates the rotational error when computing the final orientation  $\mathbf{q}_f$  with the histogram filter. To compare against a particle filter implementation, we performed 20 runs of it per object using the same number of particles as bins in the histogram filter, and the same measurement model. The average of these 20 runs is shown as the orange line.

The methods BMM M+R and BMM SMC are two different implementations of a Bayesian sequential orientation fusion based on BMMs, i.e. Bingham mixture models (Riedel et al. 2016). The Bingham distribution is a probabilistically sound, parametric distribution over orientations in 3D (Bingham 1974). In the BMM M+R method the state as well as the measurement model is represented by a Bingham mixture and therefore the Bayesian recursion can be implemented



**Fig. 10** Evaluation and comparisons for three objects *filter*, *pressure* and *control*. It is shown that the rotational error for the final orientation is generally the smallest when using the histogram filter. Note that due

analytically via multiplication of the prior and measurement mixtures (M) and a reduction (+R) of the number of mixture components of the posterior mixture. The measurement model has two mixture components:

1. A unimodal component peaked around the detected object orientation with a small and fixed uncertainty, and a mixture weight of 95%;
2. A uniformly distributed component to cope with outlier measurements (see e.g. filter object) with a mixture weight of 5%.

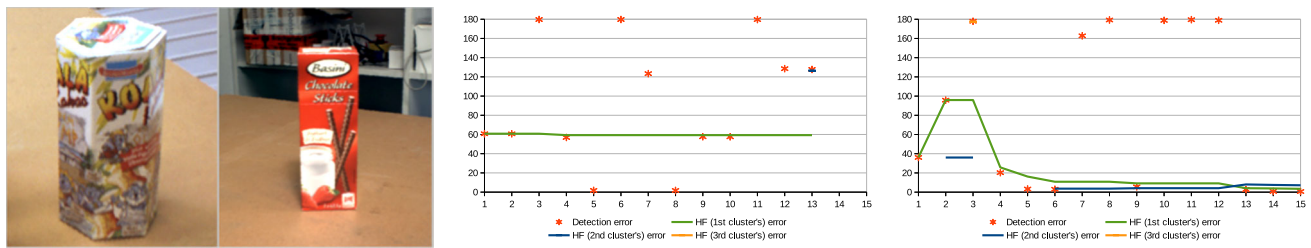
In the BMM SMC method, the same two-component measurement model is used (slightly different relative component weights of 90% to 10%), but in a standard Sequential Monte Carlo (SMC) fusion framework. The BMM measurement model serves as observation model which provides weights to the propagated orientation samples ( $N = 100,000$  samples were used by the BMM SMC approach) before they are resampled to form the posterior distribution. For more details on these Bayesian methods based on BMMs please see Riedel et al. (2016).

As shown, the histogram filter outperforms both the simple nearest neighbor clustering and Bayesian approaches for each of the objects. The peak detection parameters chosen in the previous subsection result in estimates that vary only slightly around ground truth (barely visible in the plots) after 5 to 13 steps.

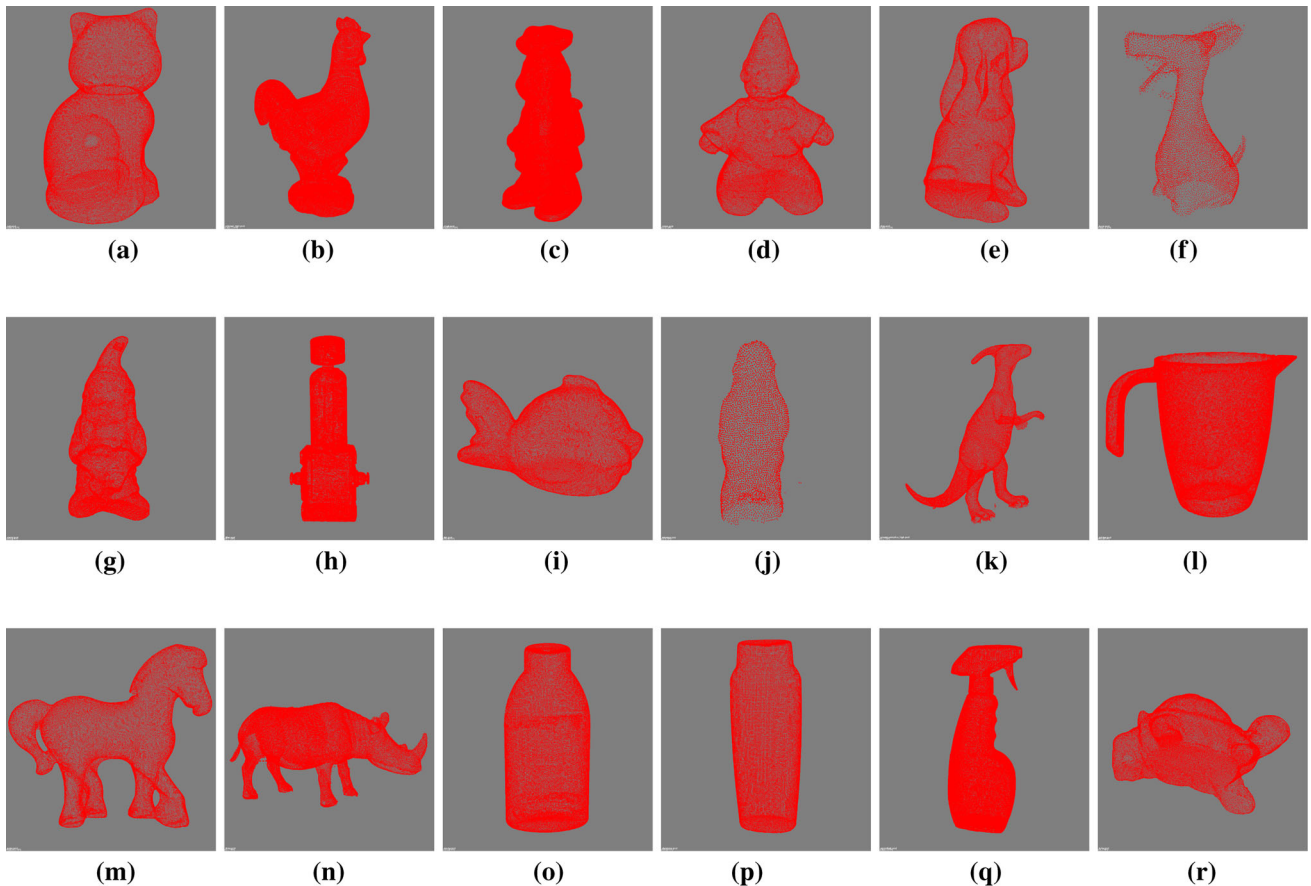
To take a closer look at the histogram filter results, we obtained additional depth images using a stereo camera of two new objects, shown in Fig. 11. The visualization of the errors of all the detected peaks shows the histogram filter converging to the correct orientation even when detection errors are very large. It must be noted that since the pose estimation method is purely depth based, all symmetrical orientations are equally likely to be detected, thus the *koala* object converging to an orientation with  $60^\circ$  error is equivalent to that of one with  $0$  (we checked that the rotations are indeed around the upright axis).

to the symmetries of the filter object some detections have very high rotational error (Fig. 9), resulting in a high average detection error (these fall above the represented range of the *left* figure)





**Fig. 11** Test objects *koala* and *chocsticks* showing different symmetry types (left images). Corresponding histogram filter orientation estimation errors in degrees, shown for all detected peaks sorted by weight (plots on the right)



**Fig. 12** The 18 objects, which have been used in the evaluation for the histogram filter. The objects show a wide range of variety, e.g. non-symmetrical objects like *pony* (m), half-symmetrical objects like *filter* (h) and symmetrical objects like *shampoo* (p). The objects f, h, and j were taken from our own work (Kriegel et al. 2013, 2015), b, c, k, and

n from the Mian dataset (Mian et al. 2006), and the remaining objects from the KIT object database (Kasper et al. 2012). a Cat. b Chicken. c Chef. d Clown. e Dog. f Dogspray. g Dwarf. h Filter. i Fish. j Santa. k Para. l Pitcher. m Pony. n Rhino. o Sauce. p Shampoo. q Spray. r Turtle

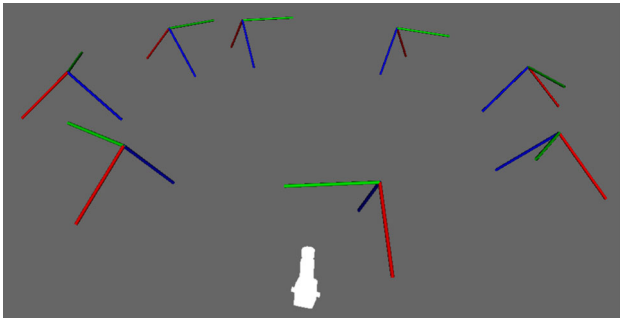
### 7.3 Large-scale evaluation

In order to evaluate the histogram filter, scans of 18 objects were used (shown in Fig. 12). Since most objects are synthetic, and in order to be able to make a large number of tests, the scans were simulated. This was done by circling around the object from a given height and the camera pointing towards the centroid of the object. In Fig. 13 some example viewpoints are shown. The viewpoints are circling around the

objects, looking at them from a  $45^\circ$  angle. A total of  $n = 30$  scans have been acquired, which equals to a rotational distance from one viewpoint to another of  $12^\circ$ .

#### 7.3.1 Results overview

In Table 2 the results for the evaluation of the histogram filter for the 18 objects from Fig. 12 are shown. In addition to the different final orientation computations introduced in



**Fig. 13** In total 30 scans have been simulated s.t. the camera pose circled above the object, pointing towards the objects centroid (8 example poses shown here)

Sect. 5.2, two non histogram filter based approaches were compared. The median of the detection error is contrasted with the error of two averaging methods, that of the mean orientation and of the intuitive KNN Clustering approach used in Kriegel et al. (2013). For 16 out of the 18 objects, the histogram filter based approaches outperform these averaging methods. For the *filter*, *sauce* and *shampoo*, a rotation  $180^\circ$  rotation around the  $z$ -axis could be considered geometrically correct (i.e. an error of  $176^\circ$  can be treated as  $4^\circ$ ).

*Mean* The mean approach gives the worst results for the final orientation's rotational error. This is because most of the objects show some kind of "symmetry", hence wrong poses

can produce relatively good overlap with the data, and thus get included in the averaging.

*KNN Clustering* The nearest neighbor clustering approach (Kriegel et al. 2013) has the best results for the non histogram filter based approach. The detected orientations  $\mathbf{q}_d$  are clustered in a  $k$ -nearest-neighbor-manner, such that, at the end there is one or a few clusters. Then, the average in the cluster with the most detections is computed. Hence, the "false" symmetrical detections are left out in the computation for the final orientation  $\mathbf{q}_f$  and the rotational error  $\Delta e(\mathbf{q}_f)$ . The results for this approach are very good, almost all of them are under  $1^\circ$  of error. It is fair to say, that this approach is a good alternative to the histogram filter based approaches.

*Voting and VW Voting* As in Sect. 5.2 described, the (verification weighted) voting approach computes the final orientations  $\mathbf{q}_f$  by voting one cell depending on the detection  $\mathbf{q}_d$  in each step, and at the end taking the corresponding representative orientation  $\mathbf{q}_r$  with the highest weight as the final orientation. When looking at the table, the results of this approach are always reasonable. However, the results are rather bad compared to the nearest neighbor clustering approach. Rotational errors around  $3^\circ$  and  $4^\circ$  are the majority. For the computation of the final orientation only once representative orientation  $\mathbf{q}_r$  is considered and therefore, the discretization errors shown in Sect. 4.2 cannot be corrected.

**Table 2** Evaluation of 18 objects with 30 scans (see Fig. 13 for the camera poses)

	Median error ( $^\circ$ )	Average orientation ( $^\circ$ )	KNN clustering ( $^\circ$ )	Voting ( $^\circ$ )	VW voting ( $^\circ$ )	CW clustering ( $^\circ$ )	CW + VW clustering ( $^\circ$ )
Cat	23.336	65.286	<b>1.342</b>	6.028	7.901	<i>1.683</i>	1.848
Chef	92.439	77.721	0.885	3.764	2.806	<i>0.710</i>	<b>0.396</b>
Chicken	1.015	21.721	<i>0.360</i>	1.939	1.939	<b>0.055</b>	<b>0.055</b>
Clown	2.047	46.423	0.114	3.553	3.553	<i>0.014</i>	<b>0.008</b>
Dog	95.431	82.736	0.197	3.764	3.764	<b>0.036</b>	<i>0.059</i>
Dogspray	157.128	96.160	0.168	163.913	163.913	<b>0.023</b>	<i>0.027</i>
Dwarf	1.445	53.975	0.154	3.764	2.663	<i>0.015</i>	<b>0.010</b>
Filter	88.570	81.026	<i>0.617</i>	178.746	178.746	<b>0.423</b>	175.674
Fish	2.291	52.354	0.237	1.551	1.547	<i>0.036</i>	<b>0.028</b>
Santa	2.260	54.285	0.391	3.764	3.764	<b>0.053</b>	<i>0.056</i>
Para	0.668	49.869	0.053	4.346	4.346	<b>0.013</b>	<i>0.020</i>
Pitcher	0.917	0.873	0.073	1.255	1.206	<i>0.012</i>	<b>0.009</b>
Pony	1.740	1.714	0.081	1.457	1.504	<b>0.018</b>	<i>0.019</i>
Rhino	1.732	42.367	0.747	1.441	1.441	<i>0.333</i>	<b>0.332</b>
Sauce	176.815	107.018	0.330	178.745	178.745	<i>0.050</i>	<b>0.048</b>
Shampoo	176.758	110.820	<b>0.512</b>	175.885	175.885	<i>176.849</i>	174.909
Spray	2.339	67.929	0.326	1.565	1.591	<i>0.050</i>	<b>0.045</b>
Turtle	3.299	44.955	0.905	3.073	1.510	<b>0.751</b>	<i>0.784</i>

The best results for each object are marked in bold, while the second best with italics. Note that a  $180^\circ$  rotation can be in some cases correct

Also, it is notable, that the verification weighted approach achieves the same results for half of the objects, for four objects it achieves worse results and only for five out of 18 objects it achieves better results.

This is the case, since most detections are correct, and if there were detections with more than 5° error, then these were symmetrical errors and not random ones.

**CW Clustering and CW+VW Clustering** The confidence weighted (and verification weighted) clustering approach achieves the best results for the histogram filter. As described in Sect. 5.2, the final orientation  $q_f$  is computed by clustering cells with a corresponding weight, which is higher than a given threshold, and afterwards computing the average of the cluster with the highest accumulated weight. This way, the discretization errors can be corrected, which have been the reason for the bad results for the voting approach. There is again no difference between the verification weighted approach and the approach without it.

### 7.3.2 Discussion

In order to visualize the results of the evaluation, a box plot for each approach has been made,<sup>5</sup> see Fig. 14.

The nearest neighbor clustering approach has the best results for the non histogram filter based approaches, with a median of 0.3° and quartiles from 0.17° to 0.6°.

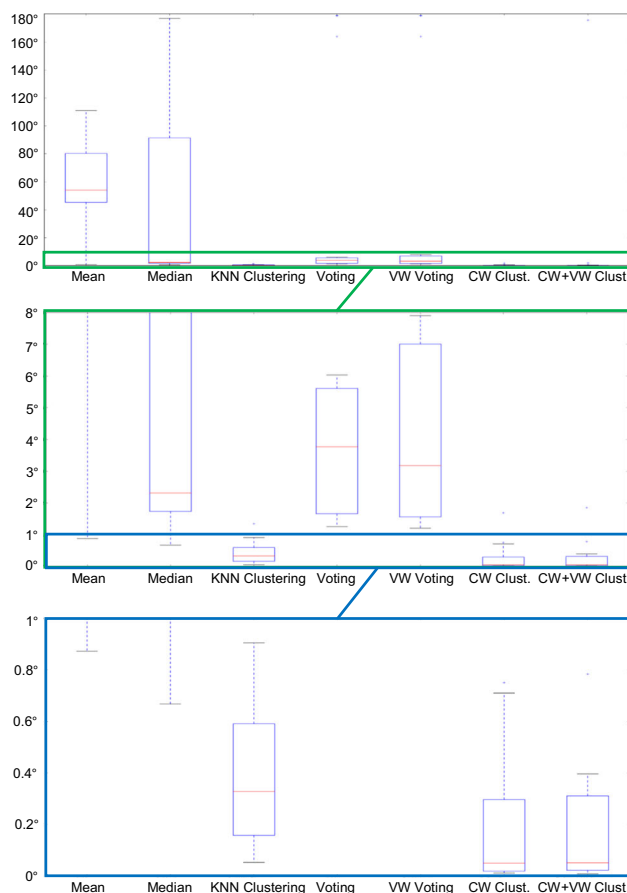
The voting based approaches of the histogram filter based ones have median errors around 3° to 4° (with relatively high variances). As it is described above, the discretization errors contribute significantly to these relatively high rotational errors, as the distance between neighboring cells can be up to 4° (see Table 1).

The confidence weighted clustering with and without verification weighting achieved the best results in this evaluation. Both of these methods have almost the same statistics. The median for both approached is at approximately 0.05°, which is by far the best result. The interquartile range is for both approximately from 0.01° to 0.03°. However, the maximum whisker for the verification weighted added approach is lower (0.4°) than the normal clustering approach (0.7°). Hence, when adding verification weighting to the confidence weighting, the variance of the results are lowered.

### 7.3.3 Incremental effects of multi-view detections

In Sect. 6.4, the estimation for NBV planning was described. There it was said, that estimating a next viewpoint is crucial to converge to a final orientation in as few steps as possible. In

<sup>5</sup> Please note that the box plot drawing library detects outliers based on interquartile range distance, marks them as dots, and excludes them from percentile calculations.

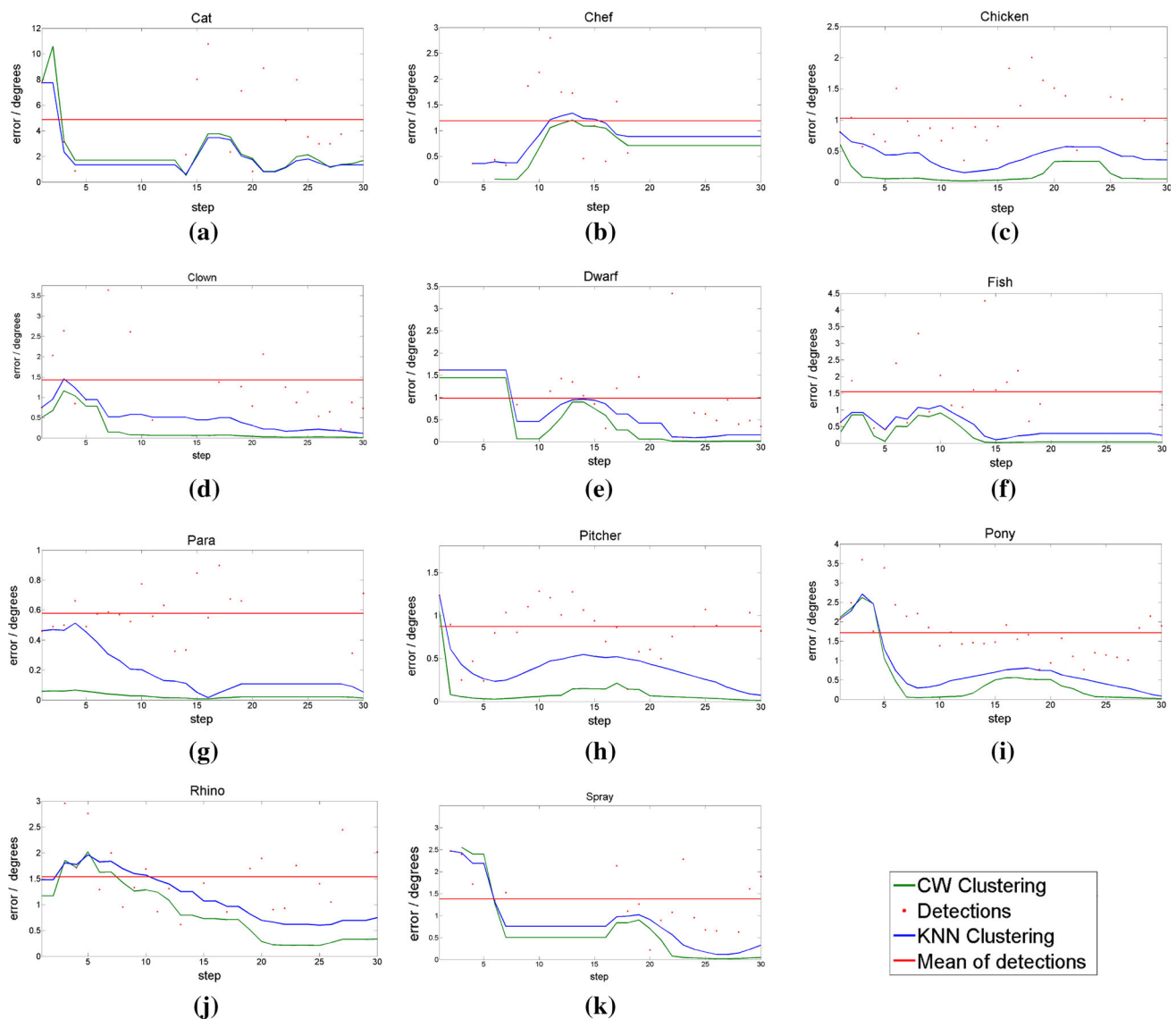


**Fig. 14** Box plots for the evaluation in Table 2. The full box plot is at the top, with zoomed in details below

this section, the step by step results of the evaluation results from Table 2, will be discussed for 11 objects, shown in Fig. 15.

For the *chicken* the final orientation seems to converge after 4 scans. But after the 16th scan, the pose estimator makes several detections in a row with a large rotational error. Hence, the final orientation’s rotational error also increases. The same applies for the *pony*, as well. The final orientation seems to converge after 7 scans, but after the 12th scan, again the final orientation’s rotational error increases due to bad detections. For the objects *rhino* or *cat* it seems, that even after 30 scans, the final orientation did not converge.

Another phenomenon is the fact, that for some objects not all detections are close to the final orientation. For example for the *chef*, the first detection, which is close to the final orientation, has been detected after four scans. And after the 18<sup>th</sup> scan, neither of the remaining 12 detections are related to the final orientation. That means, that only half of the detections are related to the final orientation. Same observations can be made for the objects *cat* and *fish*, as well.



**Fig. 15** Step by step rotational errors for the detected orientations  $\mathbf{q}_d$  and the final orientation  $\mathbf{q}_f$ , shown for a subset of the evaluated objects (the *filter* and *turtle* datasets will be presented in more detail in Fig. 18). **a** Cat. **b** Chef. **c** Chicken. **d** Clown. **e** Dwarf. **f** Fish. **g** Para. **h** Pitcher. **i** Pony. **j** Rhino. **k** Spray

These results show that having a NBV estimator is crucial, so that the final orientation converges as fast as possible, as evaluated in Sect. 7.5.

#### 7.4 Contingency table as measurement model

In Sect. 6 the estimation and the usage of the contingency table as the measurement model was described. Here we present the evaluation, when setting  $k = 8$  and the number of valid orientations being 8480. In Table 3 the evaluation of the histogram filter when using the contingency table as the measurement model for three example objects is shown. The rotational errors for the confidence weighted clustering approach are all around  $2^\circ$  to  $5^\circ$ . These errors are higher than

the errors in the evaluation in Sect. 7.3 because the evaluation there has been performed with  $k = 32$ . However, when using the contingency table as the measurement model, the rotational errors are all better compared to the rotational errors when using the confidence weighted clustering.

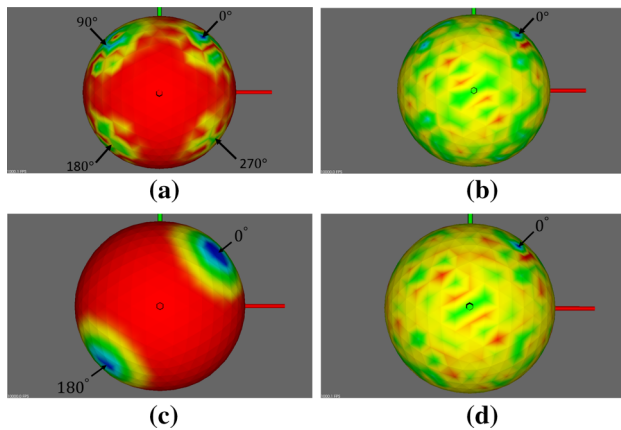
The differences between the reweighting approaches are shown in Fig. 16 for the two symmetrical objects. In Fig. 16a the final step of the confidence weighted approach is shown for the object *filter*. The reweighting with the contingency table is shown in Fig. 16b. With the confidence weighted approach multiple peaks are established, at  $0^\circ$ ,  $90^\circ$ ,  $180^\circ$  and at  $270^\circ$  around the  $z$ -axis respectively, while for the contingency table weighted approach only one peak at identity is established. The four peaks for the confidence weighted



**Table 3** Contingency Table based reweighting results for three objects compared to (verification) voting and confidence (and verification) based approaches

	Voting (°)	VW voting (°)	CW clust. (°)	CW + VW clust. (°)	CT clust. (°)
Filter	173.38	173.37	2.18	<i>2.16</i>	<b>1.63</b>
Shampoo	173.38	173.38	4.55	<i>4.24</i>	<b>3.46</b>
Turtle	2.72	2.91	3.53	3.54	<b>1.71</b>

The best results for each object are marked in bold, while the second best with italics



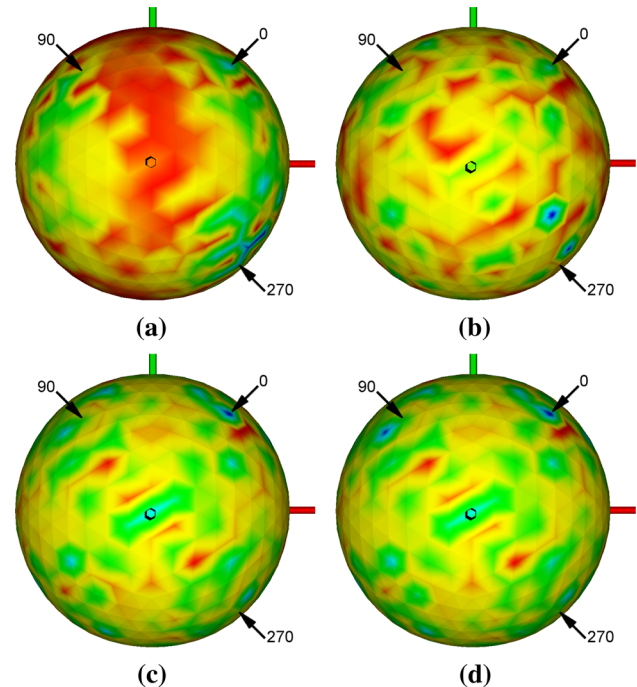
**Fig. 16** Difference between confidence weighted (a, c) and contingency table weighted (b, d) approach for the *filter* (top) and *shampoo* (bottom) objects. With CW-based reweighting four, respectively two, peaks are established (according to object symmetries), while using the CT-based reweighting only the correct one. **a** Confidence weighted based reweighting. **b** Contingency table based reweighting. **c** Confidence weighted based reweighting. **d** Contingency table based reweighting

approach are obtained since the object has symmetrical structures. The contingency table can compensate those pose estimator's errors, hence, at the end there is only one peak in the histogram filter. The same applies for the object *shampoo* (Fig. 16c, d), and on the real scans from Fig. 9 there is a similar effect (Fig. 17).

As shown, using the contingency table as the measurement model improves the computation of the final orientation, however, this has to be further evaluated.

### 7.5 Viewpoint selection

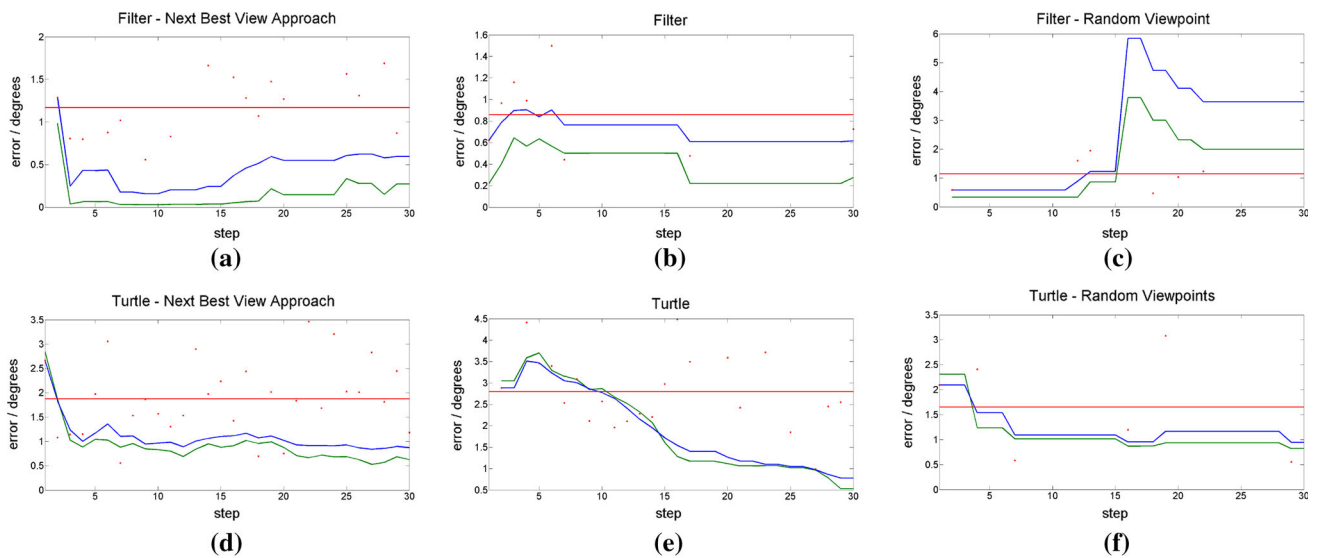
In Sect. 6.4, the NBV estimation was introduced and described. In this section, the NBV estimation approach will be evaluated. Figure 18 shows the step by step rotational errors for the detected orientations  $\mathbf{q}_d$  and the final orientation  $\mathbf{q}_f$  evaluated with different approaches. The NBV approach to estimate the next viewpoint is compared to circular (Fig. 13) and random viewpoints for the *filter* and *turtle* objects. The NBV converges fastest, while the random approach performs worst.



**Fig. 17** Histogram filter based (H.F.) results using estimated error filter model. The used filter object has four similar sides (perpendicular to each other and parallel to the  $z$ -axis). Due to object symmetries and pose estimator bias, the distributions show multiple peaks (roughly  $90^\circ$  rotated around the  $z$ -axis) at the first and last H.F. step. The correct solution is  $0^\circ$  (a maxima starting with step 12). The maximum values (blue) are 0.61, 0.47 and 0.45%, respectively in the H.F. results. **a** 20 raw detections. **b** H.F. results on first frame. **c** H.F. results after frame 12. **d** H.F. results after frame 20 (Color figure online)

## 8 Conclusions and future work

The presented histogram filter based approach improved over the individual detections even if their errors were very high, and it compared favorably to other alternative fusions approaches. Additionally, the contingency table was introduced as a measurement model, which performed better than the standard Gaussian error function. Similarly to the contingency table, the symmetry model has been computed. Using this symmetry model, the final orientation of the object can be computed considering all the symmetries of the object. Furthermore, a NBV estimator was implemented that uses the contingency table of the object to find the “least confusing” next view direction. This approach results in a faster



**Fig. 18** Step by step rotational errors for the detected orientations  $\mathbf{q}_d$  and the final orientation  $\mathbf{q}_f$  when using the NBV approach to estimate the next viewpoint compared to circular and random viewpoints. Please note the different ranges of the plots, especially the non-zero minimum

in e. See legend in Fig. 15 for more details. **a** NBV Approach for object *filter*. **b** Circular Viewpoints for object *filter*. **c** Random Viewpoints for object *filter*. **d** NBV approach for object *turtle*. **e** Circular Viewpoints for object *turtle*. **f** Random Viewpoints for object *turtle*

convergence of the final orientation than using random (or circular) viewpoints.

The proposed NBV estimation is a very efficient (constant-time lookup), but rather heuristic approach, where the NBV is the viewpoint from which the pose estimator makes the fewest confusions for the given object. It would be preferable to have a system which is based on an information-theoretic computation that balances an estimate of the expected gain from each observation as well as its computational cost. In Gao and Koller (2011), the authors present an active classification process at the test time, where each classifier in a large ensemble is viewed as a potential observation that might inform the classification process. The expected classification gain is computed using a probabilistic model that uses the outcome from previous observations. In our case, the goal would be to predict the pose estimation information gain from the next-best-view, given the CT.

**Acknowledgements** The authors would like to thank Anas Al-Nuaimi for helpful discussions, and the help of Laura Beckmann with the real-world ground truth for the evaluation.

## References

- Aldoma, A., Marton, Z. C., Tombari, F., Wohlkinger, W., Potthast, C., Zeisl, B., et al. (2012). Tutorial: Point cloud library—three-dimensional object recognition and 6 DoF pose estimation. *IEEE Robotics & Automation Magazine*, *19*(3), 80–91.
- Arbel, T., & Ferrie, F. P. (2001). Entropy-based gaze planning. *Image and Vision Computing*, *19*(11), 779–786.

- Barequet, G., & Sharir, M. (1994). Partial surface and volume matching in three dimensions. *IEEE Transactions PAMI*, *19*, 929–948.
- Barequet, G., & Sharir, M. (1999). Partial surface matching by using directed footprints. *Computational Geometry*, *12*, 45–62.
- Bingham, C. (1974). An antipodally symmetric distribution on the sphere. *The Annals of Statistics*, *2*(6), 1201–1225. doi:10.1214/aos/1176342874.
- Chen, H., & Bhanu, B. (2007). 3d free-form object recognition in range images using local surface patches. *Pattern Recognition Letters*, *28*(10), 1252–1262. doi:10.1016/j.patrec.2007.02.009.
- Chen, S., Li, Y., & Kwok, N. M. (2011). Active vision in robotic systems: A survey of recent developments. *IJRR*, *30*(11), 1343–1377.
- Chen, Y., & Medioni, G. (1992). Object modelling by registration of multiple range images. *Image and vision computing*, *10*(3), 145–155. doi:10.1016/0262-8856(92)90066-C.
- Denzler, J., & Brown, C. M. (2002). Information theoretic sensor data selection for active object recognition and state estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(2), 145–157.
- Doya, K., Ishii, S., Pouget, A., & Rao, R. P. N. (2007). *Bayesian brain: Probabilistic approaches to neural coding*. Cambridge: The MIT Press.
- Drost, B., Ulrich, M., Navab, N., & Ilic, S. (2010). Model globally, match locally: Efficient and robust 3d object recognition. In *IEEE conference on computer vision and pattern recognition (CVPR)*, 2010, pp. 998–1005. doi:10.1109/CVPR.2010.5540108
- Eidenberger, R., Grundmann, T., Feiten, W., & Zoellner, R. (2008). Fast parametric viewpoint estimation for active object detection. In *IEEE international conference on multisensor fusion and integration for intelligent systems*, 2008 (pp. 309–314). IEEE.
- Eidenberger, R., Grundmann, T., Schneider, M., Feiten, W., Fiegert, M., Wichert, G.v., et al. (2012). Scene analysis for service robots. In *Towards service robots for everyday environments* (pp. 181–213). Springer. <http://www.springerlink.com/index/N9212641455202J2.pdf>
- Eidenberger, R., Grundmann, T., & Zoellner, R. (2009). Probabilistic action planning for active scene modeling in contin-

- uous high-dimensional domains. In *IEEE international conference on robotics and automation*, 2009 (pp. 2412–2417). Ieee. doi:[10.1109/ROBOT.2009.5152598](https://doi.org/10.1109/ROBOT.2009.5152598). <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5152598>
- Fitzgibbon, A. (2001). Robust registration of 2D and 3D point sets. In: *Proceedings of the British Machine Vision Conference* (pp. 662–670). Manchester, UK.
- Gao, T., & Koller, D. (2011). Active classification based on value of classifier. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira & K. Q. Weinberger (Eds.), *Advances in neural information processing systems 24 (NIPS)* (pp. 1062–1070). Curran Associates, Inc. <http://papers.nips.cc/paper/4340-active-classification-based-on-value-of-classifier.pdf>.
- Glover, J., Rusu, R., & Bradski, G. (2011). Monte Carlo pose estimation with quaternion kernels and the bingham distribution. In *Proceedings of robotics: Science and systems*. Los Angeles, CA, USA.
- Grzyb, B. J., Castelló, V., & del Pobil, A. P. (2012). Reachable by walking: Inappropriate integration of near and far space may lead to distance errors. In: Szufnarowska, J. (Ed.), *Proceedings of the post-graduate conference on robotics and development of cognition* (pp. 12–15).
- Hough, P. (1962). Method and means for recognizing complex patterns. U.S. Patent 3.069.654.
- Kasper, A., Xue, Z., & Dillmann, R. (2012). The KIT object models database: An object model database for object recognition, localization and manipulation in service robotics. *IJRR*, 31(8), 927–934.
- Kriegel, S., Brucker, M., Marton, Z. C., Bodenmüller, T., & Suppa, M. (2013). Combining object modeling and recognition for active scene exploration. In *IEEE international conference on intelligent robots and systems (IROS)*, Tokyo, Japan.
- Kriegel, S., Rink, C., Bodenmüller, T., & Suppa, M. (2015). Efficient next-best-scan planning for autonomous 3d surface reconstruction of unknown objects. *JRTIP*, 10(4), 611–631.
- Laporte, C., & Arbel, T. (2006). Efficient discriminant viewpoint selection for active bayesian recognition. *International Journal of Computer Vision*, 68(3), 267–287. doi:[10.1007/s11263-005-4436-9](https://doi.org/10.1007/s11263-005-4436-9).
- Lynott, D., & Connell, L. (2009). Modality exclusivity norms for 423 object properties. *Behavior Research Methods*, 41(2), 558–564.
- Marton, Z. C., Pangercic, D., Blodow, N., & Beetz, M. (2011). Combined 2D–3D categorization and classification for multimodal perception systems. *The International Journal of Robotics Research*, 30(11), 1378–1402.
- Marton, Z. C., Seidel, F., Balint-Benczedi, F., & Beetz, M. (2012). *Ensembles of strong learners for multi-cue classification*. Pattern Recognition Letters (PRL), Special Issue on Scene Understandings and Behaviours Analysis.
- Mian, A., Bennamoun, M., & Owens, R. (2010). On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes. *International Journal of Computer Vision*, 89(2–3), 348–361. doi:[10.1007/s11263-009-0296-z](https://doi.org/10.1007/s11263-009-0296-z).
- Mian, A. S., Bennamoun, M., & Owens, R. (2006). Three-dimensional model-based object recognition and segmentation in cluttered scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10), 1584–1601. doi:[10.1109/TPAMI.2006.213](https://doi.org/10.1109/TPAMI.2006.213).
- Novatnack, J., & Nishino, K. (2008). Scale-dependent/invariant local 3d shape descriptors for fully automatic registration of multiple sets of range images. *ECCV*, 3, 440–453.
- Papazov, C., Haddadin, S., Parusel, S., Krieger, K., & Burschka, D. (2012). Rigid 3d geometry matching for grasping of known objects in cluttered scenes. *The International Journal of Robotics Research*, 31(4), 538–553.
- Prnobis, A., Mozos, O. M., Caputo, B., & Jensfelt, P. (2010). Multimodal semantic place classification. *The International Journal of Robotics Research (IJRR)*, 29(2–3), 298–320. doi:[10.1177/0278364909356483](https://doi.org/10.1177/0278364909356483).
- Riedel, S., Marton, Z. C., & Kriegel, S. (2016). Multi-view orientation estimation using Bingham mixture models. In: *2016 IEEE international conference on automation, quality and testing, robotics (AQTR)*. doi:[10.1109/AQTR.2016.7501381](https://doi.org/10.1109/AQTR.2016.7501381)
- Rink, C., Kriegel, S., Seth, D., Denninger, M., Marton, Z. C., & Bodenmüller, T. (2016). Monte Carlo registration and its application with autonomous robots. *Journal of Sensors*. doi:[10.1155/2016/2546819](https://doi.org/10.1155/2016/2546819)
- Rink, C., Marton, Z. C., Seth, D., Bodenmüller, T., & Suppa, M. (2013). Feature based particle filter registration of 3D surface models and its application in robotics. In *IEEE international conference on intelligent robots and systems (IROS)*, Tokyo, Japan.
- Romea, A., & Srinivasa, S. (2010). Efficient multi-view object recognition and full pose estimation. In *2010 IEEE international conference on robotics and automation (ICRA 2010)*.
- Roy, S. D., Chaudhury, S., & Banerjee, S. (2004). Active recognition through next view planning: A survey. *Pattern Recognition*, 37(3), 429–446.
- Rusu, R., Blodow, N., & Beetz, M. (2009). Fast point feature histograms (fpfh) for 3d registration. In *IEEE international conference on robotics and automation, 2009. ICRA '09* (pp. 3212–3217). doi:[10.1109/ROBOT.2009.5152473](https://doi.org/10.1109/ROBOT.2009.5152473)
- Rusu, R., Blodow, N., Marton, Z., & Beetz, M. (2008). Aligning point cloud views using persistent feature histograms. In *IEEE/RSJ international conference on intelligent robots and systems, 2008. IROS 2008* (pp. 3384–3391). doi:[10.1109/IROS.2008.4650967](https://doi.org/10.1109/IROS.2008.4650967)
- Selinger, A., & Nelson, R. (2001). Appearance-based object recognition using multiple views. In *IN CVPR01* (pp. 905–911).
- Sharf, I., Wolf, A., & Rubin, M. (2010). Arithmetic and geometric solutions for average rigid-body rotation. *Mechanism and Machine Theory*, 45(9), 1239–1251. doi:[10.1016/j.mechmachtheory.2010.05.002](https://doi.org/10.1016/j.mechmachtheory.2010.05.002). <http://www.sciencedirect.com/science/article/pii/S0094114X10000790>.
- Sipe, M. A., & Casasent, D. (2002). Feature space trajectory methods for active computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12), 1634–1643. doi:[10.1109/TPAMI.2002.1114854](https://doi.org/10.1109/TPAMI.2002.1114854).
- Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic robotics*. Cambridge, MA: MIT Press.
- Tombari, F., & Di Stefano, F. (2012). Hough voting for 3d object recognition under occlusion and clutter. *IPSJ Transactions on Computer Vision and Applications*, 4, 20–29.
- Tombari, F., Salti, S., & Luigi, D. (2010). Unique signatures of histograms for local surface description. In *Proceedings of the 11th European conference on computer vision conference on computer vision: Part III, ECCV'10* (pp. 356–369). Springer, Berlin. <http://dl.acm.org/citation.cfm?id=1927006.1927035>
- Vikstén, F., Söderberg, R., Nordberg, K., & Perwass, C. (2006). Increasing pose estimation performance using multi-cue integration. In *ICRA* (pp. 3760–3767). IEEE. <http://dblp.uni-trier.de/db/conf/icra/icra2006.html#VikstenSNP06>
- Voit, M., & Stiefelhagen, R. (2006). A bayesian approach for multi-view head pose estimation. In *IEEE international conference on multisensor fusion and integration for intelligent systems - MFI06*. Heidelberg, Germany.
- Wahl, E., Hillenbrand, U., & Hirzinger, G. (2003). Surflet-pair-relation histograms: A statistical 3D-shape representation for rapid classification. In *3DIM* (pp. 474–481).





**Zoltán Csaba Márton** is a senior researcher at the German Aerospace Center (DLR) since 2012, working on semantic perception for robotic manipulation. He received his Ph.D. degree (summa cum laude) in 2012 from the Technische Universität München (TUM), where he worked as a researcher since 2007. He is a graduate of the Technical University of Cluj Napoca, where he obtained a Dipl. Ing. (M.Sc. equivalent) degree in control engineering.



**Serkan Türker** is a Software Engineer at the Munich based Startup NavVis. He received his Master degree in Electrical Engineering and Information Technology from the Technical University of Munich in 2014 while doing his Master's Thesis with the DLR. Right after, he did an internship for one year at Robert Bosch RTC in Palo Alto doing research in the fields of Semi-Automatic Room Reconstruction from Point Clouds and Semantic Point Cloud Interpretation.

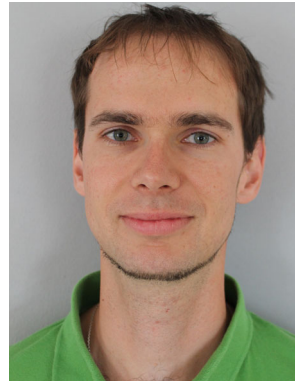
Since 2015, he is working at NavVis where he is responsible for Large Scale Point Cloud Processing and Probabilistic Point Clouds.



**Christian Rink** received his Dipl. Math. degree in 2004 from the Technical University of Munich and his M.Sc. in statistics from the Ludwig-Maximilian-Universität Munich. Since 2007 he is with the Institute of Robotics and Mechatronics, where he works in the field of mobile robots and 3D-modeling. The focus of his interests is in sequential pose estimation.



**Manuel Brucker** received his Dipl.-Inf. degree in computer science from the University of Erlangen-Nürnberg in 2011. Since then, he has been working as a research assistant at the DLR's Institute of Robotics and Mechatronics. His research interests are pattern recognition, computer vision, computer graphics, object recognition and object localization.



**Simon Kriegel** is a senior researcher at the Department of Perception and Cognition at the Robotics and Mechatronics Center, German Aerospace Center (DLR). From the Technical University of Munich, he received his Dipl.-Ing. degree in Information Technology in 2008 and Ph.D. in computer science in 2015. He started his career at KUKA Roboter GmbH, where he worked on different research projects in the field of object detection and manipulation for unloading tasks with industrial robots. Since 2009 he is with DLR where his research focuses on 3D modeling, Next-Best-View planning and exploration.



**Tim Bodenmüller** is a senior researcher at the Department of Perception and Cognition at the Robotics and Mechatronics Center, German Aerospace Center (DLR). He received his Dipl.-Ing. degree in electrical and information engineering in 2001 from the Technical University of Darmstadt and his Ph.D. in electrical engineering in 2009 from the Technical University of Munich. Since 2001, he is with the Institute of Robotics and Mechatronics, where he works in the field of 3D-sensing, -modeling, and -processing.



**Sebastian Riedel** received his M.Sc. in Computer Science from the Technical University Munich in early 2015. Since mid 2015 he works as researcher at the Department of Cognitive Robotics at the Robotics and Mechatronics Center, German Aerospace Center (DLR). Among his current research interests are experience- and data-driven approaches to fault detection and task execution monitoring as well as approaches for automatic and context-dependent algorithm parametrization.