Knowing Your Limits – Self-Evaluation and Prediction in Object Recognition

Michael Zillich, Johann Prankl, Thomas Mörwald, Markus Vincze

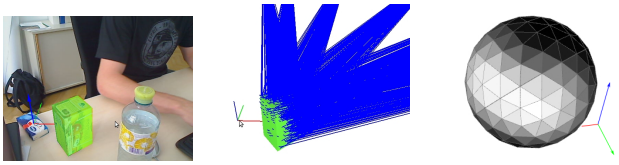
***Abstract*— Allowing a robot to acquire 3D object models autonomously not only requires robust feature detection and learning methods but also mechanisms for guiding learning and assessing learning progress. In this paper we present probabilistic measures for observed detection success, predicted detection success and the completeness of learned models, where learning is incremental and online. This allows the robot to decide when to add a new keyframe to its view-based object model, where to look next in order to complete the model, predicting the probability of successful object detection given the model trained so far as well as knowing when to stop learning.**

I. INTRODUCTION

Autonomous learning robots often face the exploration-exploitation dilemma in one form or another: should I continue learning (exploration) or make use of the models learned so far (exploitation)? Or put another way, when have I learned enough in order to satisfactorily complete my tasks? This means that not only does the robot need to represent its knowledge but also the limits of its knowledge. This becomes all the more important in integrated robotic systems which have to make decisions based on observations drawn from a multitude of modalities. All of these observations will carry some degree of uncertainty, and it is paramount that these uncertainties are formulated consistently in order to support well informed decisions. Moreover, the robot should be enabled to improve performance in future tasks by reducing uncertainties, i.e. it should be able to derive actions that reduce gaps in its knowledge. Again this increase in knowledge must be formulated consistently to allow planning for optimal knowledge gathering actions.

Within the scope of this paper we are concerned with learning 3D object models for recognition. Object recognition has made impressive advances and increasingly powerful methods have shown their applicability in challenging scenes [1], [2], [3]. The focus of research however often lies in optimising recognition (robustness, speed, generality) while the learning phase is typically done offline, i.e. outside the robot's normal execution of tasks. Consider an autonomous robot with tasks including recognition of objects such as fetching various items, where these items are not known in advance. So one of the tasks of the robot, e.g. while not being occupied with more urgent things, will be to wander about and learn new objects which might feature as part of a fetching task at a later time.

Michael Zillich, Johann Prankl, Thomas Mörwald and Markus Vincze are with the Automation and Control Institute, Vienna University of Technology, Austria {zillich, prankl, moerwald, vincze}@acin.tuwien.ac.at



(a) (b) (c)

Fig. 1. Learning an object model: The object in the scene (left) and bundles of features with their view vectors (in blue) after acquiring some views of the object. View sphere (right) with brighter shades of grey indicating that the object has been learned from the respective direction.

In this paper we explicitly concentrate on the process of acquiring these object models. This requires representing the completeness of models acquired so far as well as mechanisms to support planning for further knowledge gathering actions. We present three probabilistic measures of *observed detection success*, *predicted detection success* and *model completeness* which allow to reason about when to extend the model, where to look next as well as predict the probability of successful detection given the model learned so far.

Fig. 1 illustrates the basic idea. Object models are based on associating SURF features [4] with 3D locations on the object's surface. Models are built online while tracking the object based on the model acquired so far. New features are mapped onto the 3D object surface and associated with the view direction from which they were captured. So we know which object views are covered so far, which is represented as "bundles" of view vectors associated with features in Fig. 1(b). The "view sphere" shown in Fig. 1(c) illustrates completeness of the model, where brighter parts indicate a higher probability of detecting the object from the respective direction. A weighted sum over the whole sphere gives the expected probability of detection and the fringes of the bright areas constitute learning opportunities.

After a review of related work, we describe learning of new object views in Section III. Section IV gives an overview of the proposed recognition approach, including building of a vocabulary tree (Section IV-A), voting for object hypotheses (Section IV-B) and estimation of the poses of multiple objects (Section IV-C). Our probabilistic measures of detection success and completeness are introduced in Section V and evaluated in Section VI.

II. RELATED WORK

Various approaches to recognise specific object instances [5], [6], [7] or object categories [8], [9] in monocular images have been proposed. These approaches are optimised for recognition in complex environments, whereas training

is done offline in a (weakly) supervised fashion. In contrast we concentrate on learning objects online while tracking the model acquired so far. There also exist approaches which optimise detection for tracking in real time [10], [11], [2]. Özuysal et al. [2] rely on randomised trees for wide baseline feature matching. Assuming the object moves slowly in the first frames this approach is able to detect the object pose to initialise the initial model. In the following frames new features are added and features which cannot be reliably found are discarded. Once all the training frames are processed, bundle-adjustment is used to refine the geometry. Özuysal et al. use simple image patches to describe the interest points, instead we rely on SURF [4] which can be computed very fast and it has been shown that these features are adequate for recognition of multiple objects in complex environments.

The approaches described in [12], [13] focus on online learning of object models to avoid manual labelling of training images. Where in [12] a robust background model, a tracker and an on-line learning method are combined, in [13] classifier-based keypoint descriptions allowing incorporation of background information are learned.

With respect to efficient interest point matching the work by Riemenschneider et al. [14] is most similar to our approach. They use a vocabulary tree originally developed by Nister et al. [15] which represents prototype descriptors in a hierarchical structure for a fast matching of interest points. Instead Lowe [5] proposes an approximate nearest-neighbour method based on a kd-tree for matching of interest points. Sivic et al. [7] propose matching of query interest points with a codebook which is weighted based on the entropy. For ranking of matched database images a Term Frequency - Inverse Document Frequency (TF-IDF) scoring is used. Likewise Nister, Philbin et al. [16] rely on a hierarchical structure and introduce a method based on randomised trees to avoid the quantisation effects from k-means clustering.

An interactive modelling approach was introduced in Pan et al. [17]. For this purpose an online Structure-from-Motion algorithm is used for camera pose tracking, and the geometry is reconstructed using tetrahedron carving on the Delaunay tetrahedralization of the point cloud. Weise et al. [18] propose an interactive 3D scanning system that allows users to scan complete object geometry by turning the object around in front of a real-time 3D range scanner. To avoid artefacts they propose online loop closure and outlier handling for model reconstruction. Both approaches aim to reconstruct the 3D geometry of objects, either by means of Structure-from-Motion or with an additional sensor that provides the 3D data.

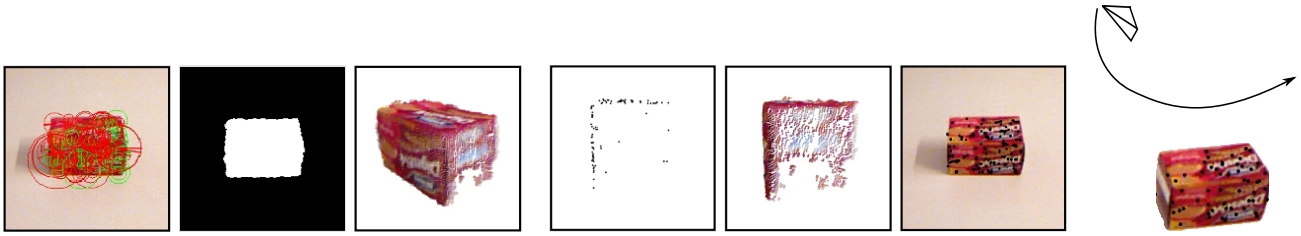
There also exist some approaches which combine object recognition and pose registration [1], [19], [3], [20], [21], [22]. Gordon and Lowe [1] build a 3D model composed of SIFT descriptors in an offline training phase by performing structure and motion estimation. The online phase then uses RANSAC to estimate 6D pose from 2D-3D correspondences. The system though is geared at augmented reality applications and the scene is not segmented into objects. Most similar to our approach is the work by Collet et al. [3]. They extend the above for application in the robotics domain, specifically by augmenting RANSAC with a Mean-Shift clustering step to allow recognition of multiple instances of the same object. The system does require manual segmentation of the object in each training image though. Furthermore the obtained sparse 3D point model has to be manually aligned with a CAD model of the object, so the whole procedure requires considerable user intervention. Instead we avoid user interaction by learning objects from salient regions popping out from a dominant ground plane. Furthermore, we combine different sensor modalities, namely the rgb-image and the point cloud acquired with the rgb-depth sensor *Kinect* to build scale corrected object models and to improve pose registration.

III. ONLINE LEARNING OF OBJECT MODELS

Currently available recognition systems rely on offline training of object models. Instead we propose to track the object model acquired so far and to add new views to the model whenever there is a lack of information. To individuate objects and separate them from the background we detect the table plane in the 3D point cloud acquired with the rgb-depth sensor *Kinect* recently developed by PrimeSense. Clusters of the point cloud which pop out from the table plane are used to build the initial object model.

In detail, first the table plane is robustly detected using RANSAC. The point cloud acquired from Kinect is organised in a 2D grid. This can be exploited for clustering the remaining 3D points. Instead of using a neighbourhood graph (e.g. the ANN [23] or FLANN [24]) we implemented a connected component analysis which directly parses the point cloud grid. To split objects which touch in the grid, but are separated in depth a cut-off threshold of the depth value is used. Then interest points within the region of interest, detected in the corresponding grey scale image are associated with their location of the object in 3D. Because of the extrinsic calibration of the rgb-camera and the depth-camera, the initial object model can simply be created by assigning the interest points to the according location in the organised point cloud. In the following frames the relative pose of the camera with respect to the tracking reference frame – i.e. the latest view added to the model – is detected. For this interest points of the tracking reference frame which have a corresponding 3D location are matched with interest points detected in the current image. Then the robust pose is computed using RANSAC [25] and the 3-point pose algorithm [26]. Additionally, we use the depth data for early pruning of RANSAC hypotheses and immediately discard pose hypotheses which are not supported by the point cloud.

To decide when to learn a new view we project the 3D model points of the currently tracked view into the image and count the number of projected points which are supported by a detected interest point, i.e., where the distance to an interest point is less than a threshold t_{inl}. Consequently, we compute



(a) Segment object using 3D point clouds and plane pop-out

(b) Map interest points to the 3D object surface

(c) Track object pose and add new keyframes to the object model

Fig. 2. Learning of interest point models. Image (a)-left shows interest points detected within the region of interest (a)-centre, which is computed from the point cloud. Image (a)-right shows the point cloud with the table plane is already removed. Note that although the images show a clean table plane the rgb-depth sensor used in our experiments handles both, highly cluttered scenes and clean untextured tables. Images (b)-left/centre indicate the alignment of interest points and 3D structure, and (b)-right shows 3D locations of the interest points projected to the object. Image (c) depicts a possible camera trajectory, where keyframes are added and successively a more complete object model is created.

the confidence values

$$c_{i,j} = \frac{n_{support,i,j}}{n_{model,i.j}}, \quad (1)$$

$$c_i = \max_j c_{i,j} \quad (2)$$

where $n_{support,i,j}$ is the number of interest points which support 3D model points and $n_{model,i,j}$ stands for the number of points of the detected view j of the object model M_i. In case c_i gets low further model views near to the current camera view are tested for support. If a model view with a higher confidence is found this becomes the new reference frame for tracking and no new view is learned. In case the confidence sinks below 0.6, indicating that almost half of the interest points of the current view are no longer visible, a new view is added to the model. However this only happens if at the same time the probability that the object was tracked correctly is high. This probability of *observed detection success* is introduced in Section V.

For learning a new view, first the camera pose is refined with non-linear optimisation using the sparse bundle adjustment implementation by Lourakis [27]. Then the 3D point locations from the object are assigned to interest points. If the interest point can be matched to a model interest point it is linked to the according 3D point, otherwise the corresponding 3D location from the point cloud is transformed to object coordinates. This new learned frame is then used as reference frame for tracking in the next frames.

Summarised, the object model consists of interest points which are organised in keyframes. Thus we propose a view-based representation using interest points which are associated with their 3D object location. To distinguish between object and background, as well as for the metric reconstruction of the interest points the point cloud from the rgb-depth sensor is used. Hence, learning is performed online while exploring the object and visiting new view points. In case viewpoints visited before are passed again recognised interest points are linked for a non-linear refinement of the structure. Furthermore, the completeness of the object model is represented by a spherical histogram. Fig. 2 depicts the different learning steps. The probabilistic formalisation for this is shown in Section V. In the next section we briefly describe how the online characteristic of model learning fits to the proposed recogniser.

IV. OBJECT RECOGNITION

For recognition of multiple objects an efficient representation of the interest points is important. In [15] Nistér et al. have shown that vocabulary trees can be used for efficient indexing of large image databases. We adapted this approach for recognition of specific objects and use the resulting ranked list of found objects for a successive pose registration. In what follows we briefly describe the basic approach and explain our modification for specific object detection.

A. Vocabulary Tree

The vocabulary tree defines a hierarchical quantisation represented by prototype interest points, which is learned offline by unsupervised k-means clustering. Instead of k defining the final number of clusters, k is the number of children of each node. Nistér has shown that for image retrieval a large vocabulary up to one million leaves improves the recognition performance. Once the tree is created the interest points of a particular image can be matched efficiently by matching against prototype descriptors of nodes and parsing down the tree. To recognise an image the task now is to compare how similar are the paths for the descriptors from a database image and a query image. To account for ambiguous descriptor assignment, weights w_i based on entropy are assigned to nodes. Thus query q_i and database vectors d_i are defined by

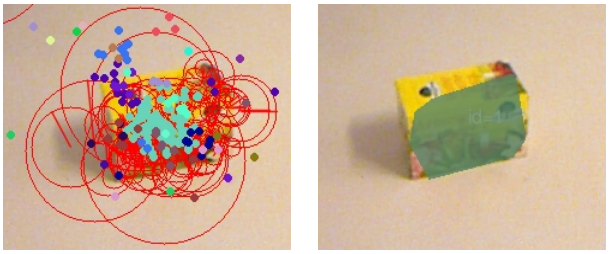
$$q_i = n_i w_i \quad (3)$$

$$d_i = m_i w_i \quad (4)$$

where n_i and m_i are the number of interest points of the query image and the database image with a path through a particular node. The score for a database image

$$s(q,d) = \| \frac{q}{\|q\|} - \frac{d}{\|d\|} \| \quad (5)$$

is then defined by the normalised difference between the query and the database vector. Likewise, as proposed by Nistér and by Sivic et al. [7] we use a *term frequency-inverse*



(a) (b)

Fig. 3. Interest points (red circles) vote for object centres (coloured dots) which results in the correct pose indicated with the green overlay (left image).

document frequency (TF-IDF) scheme. Thus the weight w_i results in

$$w_i = \ln \frac{N}{N_i}. \tag{6}$$

In contrast to Nistér, who query for images in a database, we want to rank whole objects. Hence, N is the number of objects in the database and N_i stands for the number of objects in the database with at least one descriptor path through node i. Scoring can be efficiently implemented with an inverted file structure, where every node holds an inverted file and stores the ID of the object, the according view in which it occurs and the term frequency m_i. Thus, according to the score $s(q, d)$ a ranked list of object hypotheses is generated. Furthermore, we directly store the ID of the interest points of the models in the leave nodes and thus we additionally get tentative matches for the query interest points and the ranked object models.

B. Voting for Object Hypotheses

The final result of our recognition approach should include the location of the object with respect to the camera. The number of iterations required for RANSAC increases with the outlier rate. Because of the vocabulary based matching approach however the outlier rate can increase up to 90% and thus it is necessary to filter the matches. We integrated a voting scheme (in image space) followed by meanshift clustering of the votes to group the most promising matches. In detail, the relative scale s_{rel} and the relative orientation θ_{rel} of the query interest point and the model interest point is used to vote for an object location

$$\mathbf{x}_v = \mathbf{R}\,\mathbf{x}_d\,s_{rel} + \mathbf{x}_q, \tag{7}$$

where the 2×2 matrix $\mathbf{R}$ stands for the rotation matrix computed from θ_{rel}, $\mathbf{x}_d$ is the location of the model interest point relative to the object centre and $\mathbf{x}_q$ is the location of the interest point in the query image. Thus, votes of correct matches accumulate at the centre of the object. To group the votes we use mean shift clustering (cp. [28]), which is an appropriate tool because of the unknown number of objects in the image.

Fig. 3(a) shows detected interest points (red circles) and the corresponding votes for object centres (coloured dots). It can be seen, that the correct mean shift cluster (cyan dots) accumulates most of the votes and thus leads to the most accurate pose indicated with the green overlay in Fig. 3(b).

Fig. 4. Typical scenario to test the object recogniser. The coloured overlays (computed from convex hulls of matched interest points) indicate the recognised object view.

C. Robust Pose Estimation of Multiple Objects

Once corresponding model and query image interest point pairs are found and good object hypotheses, represented by clusters of point pairs, are formed the accurate pose can be estimated. Likewise, as described in Section III we use RANSAC and the 3-point pose algorithm. In each iteration a pose hypothesis for each cluster is computed. If the new pose hypothesis of a cluster is a better explanation than the according stored pose, i.e., the number of supporting point pairs is larger, it is replaced. The algorithm stops if the probability

$$\eta = (1 - \epsilon^n)^k \leq \eta_0, \tag{8}$$

that no correct set of hypotheses is found is smaller than the desired failure rate η_0. The true inlier ratio ϵ is estimated with the best hypothesis available up to now. In contrast to typical RANSAC approaches, where object models are computed sequentially we estimate a pose for each mean shift cluster. Hence, the inlier ratio $\epsilon = z_{explained}/z_{matched}$ is the number of interest points which are explained at least by one object hypothesis divided by the number of interest points which have a minimum of one tentative match.

Fig. 4 shows a typical test scenario where more than ten objects are recognised at the same time.

V. REPRESENTING COMPLETENESS

While being a valid and intuitively clear indicator for the quality of a detection outcome and lying in the range $[0, 1]$, the confidence defined in Eq. 2 is not a probability. Making informed decisions based on such a value is difficult, especially when fusing detection results with other observations within a complete robotic system. Simple thresholding in order to force crisp outcomes of "found" and "not-found" leads to brittle systems. This section presents a sound probabilistic notion of detection success, as well as a measure of model completeness.

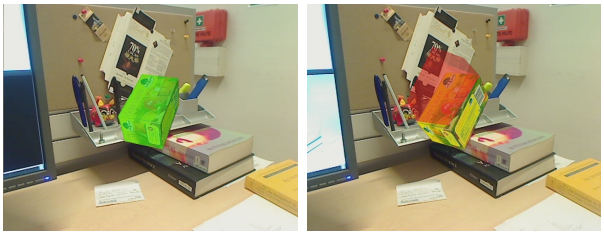


Fig. 5. Rendered virtual views of an object for evaluating detection probability, with a true positive (left) and a true negative (right).

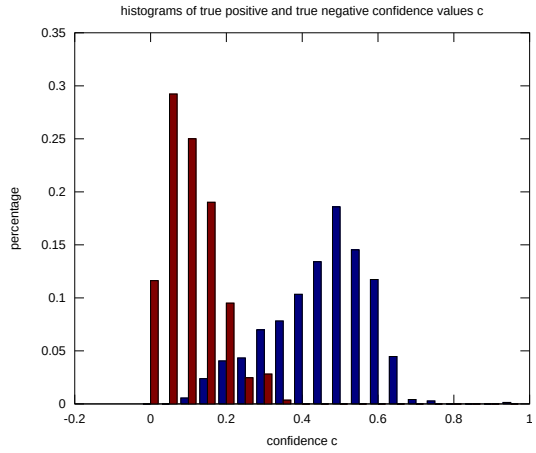


Fig. 6. Histograms for true positives (blue) and true negatives (red)

A. Probability of Detection

Taking an approach from multi-view object detection by [29], [30], [31] we define a generative detector model

$$\begin{aligned} p(c \,|\, o = true) \\ p(c \,|\, o = false) \end{aligned} \tag{9}$$

I.e. we model the probability of detection confidence given that we have a true positive or a true negative respectively. E.g. for the confidence as defined in Eq. 2 it turns out that a confidence of 0.4 typically already indicates almost absolute certainty of having a successful detection, while confidences under 0.1 tend to be false positives. We obtain training examples by transforming a virtual object model with 1000 random rotations, 252 scales and varying levels of artificial noise and blur. These virtual training examples can be created during the learning phase, though in our case they were generated beforehand for later comparison to ground truth. Fig. 5 shows examples of a true positive (in green) and true negative (in red). The threshold for accepting a detection result as a positive example was set to $4\,cm$ of distance deviation (using the error measure proposed in Section IV-B in [32]), where we found that changing the threshold had no significant impact on results. Moreover evaluation (Section VI) further justified this particular value.

Fig. 6 shows examples of histograms of confidence values for true positives and true negatives for one of our objects. As can be seen we obtain monomodal distributions (if we did not this would indicate a badly chosen confidence measure) and we fit two Gaussians, as shown in Fig. 7 Following Bayes rule we can then infer the posterior probability of a

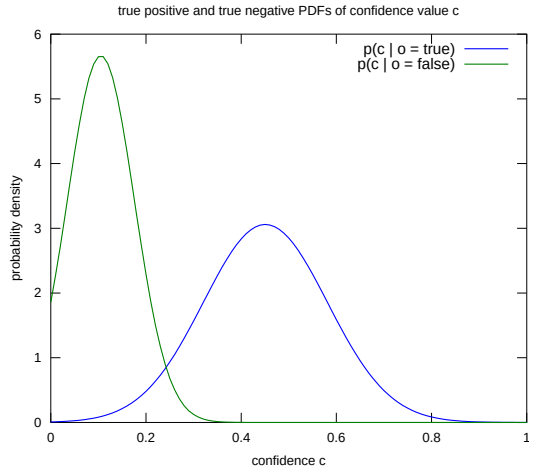


Fig. 7. Estimated Gaussian PDFs of confidence for true positives (blue) and true negatives (green)

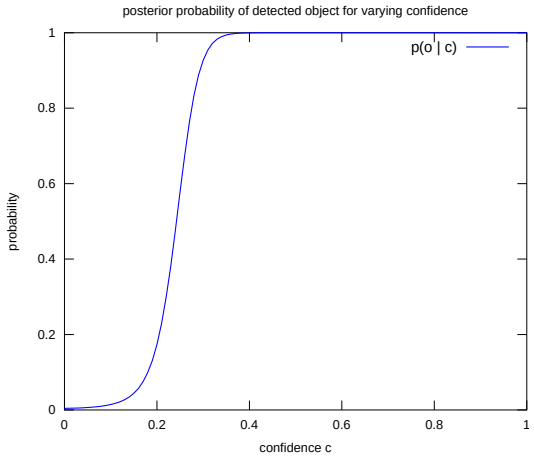


Fig. 8. *Observed detection success*: Posterior probability of detected object for given confidence

detected object

$$\begin{aligned} p(o \,|\, c) &= \frac{p(c \,|\, o)\ p(o)}{p(c)} \\ &= \frac{p(c \,|\, o)\ p(o)}{\sum_{k \in \{t,f\}} p(c \,|\, o = k)} \end{aligned} \tag{10}$$

where the prior for detecting an object could come from context knowledge such as current room category or bottom up attention, and was set to 1 for all objects in our experiments. Eq. 10 is used during learning in Section III to decide whether tracking of the current view is still reliable. Furthermore is returned to the user by the recogniser as a measure of *observed detection success*.

Fig. 8 shows the posterior probability of having successfully detected an object given detector confidence for the training data of Fig. 6.

B. Representing Completeness

In order to obtain a quantitative measure of completeness we need to know the probability of detecting the object given the views learned so far. To this end we learn the probability of detecting an object view for a given out of plane rotation

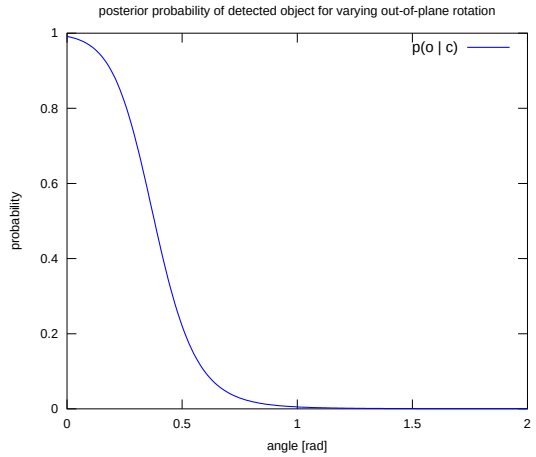


Fig. 9. *Predicted detection success*: Posterior probability of detected object for given rotation, for a single learned view.

θ. As in Section V-A we start with labelled training data, to learn

$$\begin{aligned} p(\theta \,|\, o_j = true) \\ p(\theta \,|\, o_j = false) \end{aligned} \tag{11}$$

i.e. the probability of observing a particular out of plane rotation θ for a detected or missed object view o_j. We use Bayes rule to get the *predicted detection success* for a given rotation

$$\begin{aligned} p(o_j \,|\, \theta) &= \frac{p(\theta \,|\, o_j)\ p(o_j)}{p(\theta)} \\ &= \frac{p(\theta \,|\, o_j)\ p(o_j)}{\sum_{k \in \{t,f\}} p(\theta | o_j = k)} \end{aligned} \tag{12}$$

Fig. 9 shows that recognition probability for a single learned view drops to 50% around $22°$, which is about the expected value for the feature descriptor used.

Note that the same procedure applies to varying scale instead of rotation, and in fact other environmental factors such as lighting which affect recognition. However it is only rotation and scale that we can actively influence.

To arrive at a measure of *model completeness* we take the expected detection probability over all learned views (note that we actually have to vary over two angles - azimuth and inclination, for notational simplicity however we use a single angle θ)

$$\hat{p}(o) = \sum_{\theta,j} p(o_j \,|\, \theta) p(\theta) \tag{13}$$

where $p(\theta)$ can take into account that certain views are less likely than others and thus possibly not even learned (such as the underside of an object).

C. Taking Action

The above view-based representation with associated detection probabilities and measure of completeness allows us to inform the system about next learning steps.

Starting with a single learned view we can choose a rotation for which the probability of detecting the object in the new view is high enough to allow associating the two views with the same object model but low enough to warrant learning the additional view. Note that our approach relies on tracking the previous most probable view in order to associate newly added views with a correct object pose. So learning can take place only at the fringe of currently available knowledge, i.e. the border between "bright" and "dark" areas in Fig. 1(c).

To arrive at a utility to drive exploration we need a gain (higher probability of detection after acquiring a new training view) as well as costs associated with reaching that view and attempting a learning step. Costs are measured in run-time and are composed of path planning for either moving the arm mounted camera or the whole platform (e.g. $0.5\,s$), executing the planned movement (several seconds) and attempting the learning step. The learning step consists of first detecting the object using the views learned so far (e.g. $0.5\,s$), which will succeed with a probability given by Eq. 12 and adding the new view. (Note that actually new views will only be added if they are sufficiently different to previous views, which is determined inside the learner using the confidence metric of Eq. 2. So there is a (small) probability that a new view will eventually not be added. But we can safely ignore these cases)

We define gain as the expected decrease in $\hat{p}(o = false)$ after learning the new $(n+1)$-th view:

$$g \quad = \quad \hat{p}_n(o = false) - \hat{p}_{n+1}(o = false) \tag{14}$$

with

$$\hat{p}_n(o = false) \quad = \quad 1 - \sum_{\theta} \sum_{j=1}^{n} p(o_j \,|\, \theta) p(\theta) \tag{15}$$

$$\hat{p}_{n+1}(o = false) \quad = \quad 1 - \sum_{\theta} \sum_{j=1}^{n+1} p(o_j \,|\, \theta) p(\theta) \tag{16}$$

I.e. we tentatively add the (empty) future view to our model together with its *predicted detection success* mode (which we assume to be the same for all views) and calculate the increase in detection probability.

VI. Results

In this section we evaluate the measures for *observed detection success* (Eq. 10) and *predicted detection success* (Eq. 12). Note that it is not our goal here to evaluate recognition performance. Nor can we at this point evaluate a complete system including *model completeness*, where learning is guided using our measures. Before doing so we first need to establish that these measures learned from virtual training data actually match ground truth obtained from real images.

To learn our measures we rendered five virtual objects into background images of realistic scenes with randomly chosen poses (see Fig. 5). We selected a random object pose of each object and trained an initial recognition model. Then we rotated the virtual object to 1000 randomly chosen poses with different viewing angles from $0..60°$ and tried to recognise the object. Furthermore we changed the scale of the object from 0.5 to 2 times the learning distance which results in 252 images for each of the 5 objects. In order to simulate

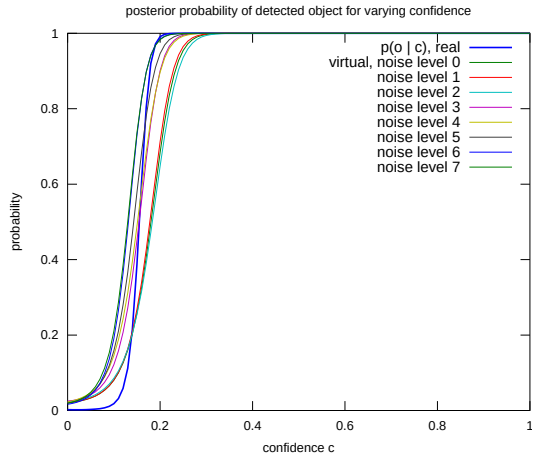


Fig. 10. Comparison of *observed detection success* learned from virtual training data including various levels of noise (thin lines) and learned from real ground truth data (thick line).

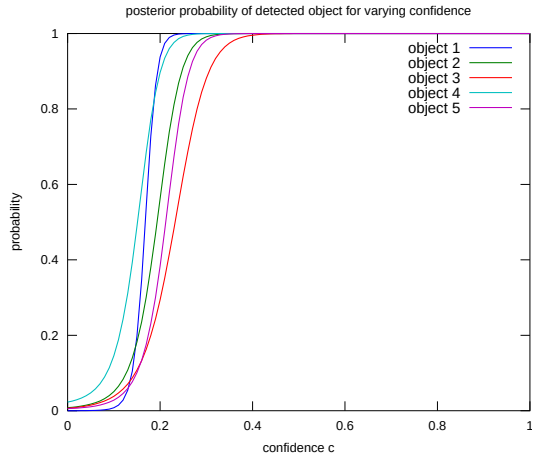


Fig. 11. Comparison of *observed detection success* for different objects.

realistic conditions we added Gaussian noise and blur at 8 levels, with $\sigma_{noise} = 0..16$ grey levels (out of 255) and blur with $\sigma_{blur} = 0..1.6$ pixels. After learning the measures for *observed detection success predicted detection success*, we evaluated them on four real sequences, where we rigidly attached a checker board pattern to the objects to create the ground truth pose data. The pose of the pattern is estimated with a standard DLT algorithm, followed by a non-linear optimisation of the pose using the sparse bundle adjustment implementation by Lourakis [27]. In total we used about 50000 virtually rendered images for learning and 461 real images for testing.

Fig. 10 shows the comparison of *observed detection success* learned from virtual vs. real ground truth training data for one of the objects. As can be seen increasing noise levels shift confidence (as is to be expected), and the true curve lies within the band defined by the various virtual curves. The real curve however has a steeper slope indicating that training from real data allows better discrimination between positives and negatives.

Fig. 11 shows the comparison of *observed detection success* for different objects. As can be seen the curves differ considerably indicating that different objects pose

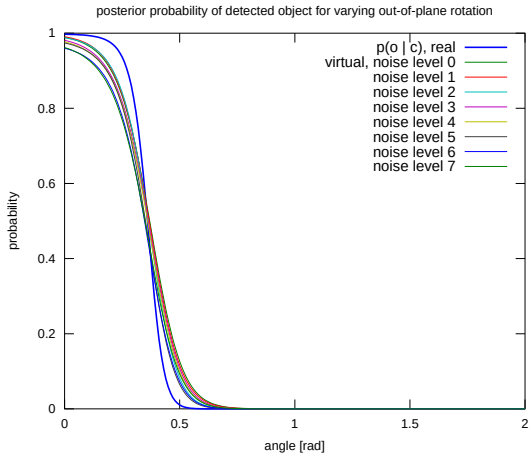


Fig. 12. Comparison of *predicted detection success* learned from virtual training data including various levels of noise (thin lines) and learned from real ground truth data (thick line).

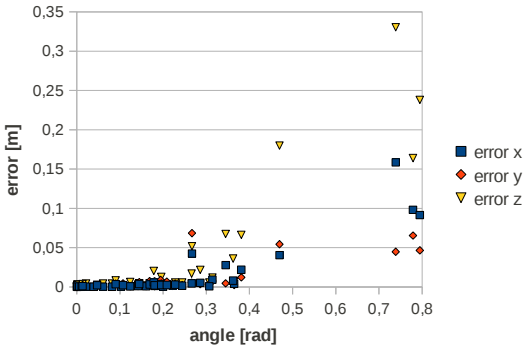


Fig. 13. Accuracy of the pose of the recognised objects for real data. The evaluation is done in camera coordinates with the z-axis pointing towards the object. Hence, *error x* and *error y* represent the deviation from ground truth within the image plane and *error z* is the depth error.

varying difficulties for recognition. This further indicates that *observed detection success* should be learned for each object individually.

Fig. 12 compares *predicted detection success* for virtual (with varying noise level) and real training data for the same object as in Fig. 10. We can see that the curves more or less intersect around $22°$ where probability drops to 50% and that again training from real data allows for better discrimination between positives and negatives.

Fig. 13 shows the accuracy of pose registration of the proposed approach for the real image sequences. For a measure of the error we again used the scheme proposed in [32]. As expected, it can be seen that the depth error is slightly higher than the error within the image plane. Furthermore, the error increases with an increasing rotation in depth θ and passes $4\,cm$ at about $20°$, after which performance rapidly degrades. This is in accordance with Fig.12 which predicts detection success to drop below 50% around $22°$. Motivated by the results of the accuracy evaluation we use this threshold of $4\,cm$ to distinguish between true positive (tp) and false positive (fp) in Section V-A.

VII. Conclusion

In this paper we considered the process of acquiring object models for recognition by an autonomous robot which needs to plan for knowledge gathering actions and reason about model completeness. We presented a view based online learning mechanism with an associated recogniser. The main contribution lies in proposing three learned probabilistic measures for *observed detection success*, *predicted detection success* and *model completeness*. Evaluation showed that learning these measures from virtual training data, which can be generated from partially learned object models during object learning, shows results comparable to learning from real ground truth data (which is of course not available during learning). These measures allow the robot to represent its knowledge of objects as well as the limit of its knowledge in a probabilistic manner compatible with probabilistic reasoning mechanisms elsewhere in the system and thus to plan for actions to extend its knowledge.

As a next step we will integrate the presented work into a complete system that can perform learning strategies based on our measure for *model completeness* and compare these to more ad-hoc or random learning strategies.

Acknowledgement

The research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement No. 215181, CogX and No. 215821, GRASP.

References

[1] I. Gordon and D. G. Lowe, "What and where: 3D object recognition with accurate pose," in *Toward Category-Level Object Recognition*, J. Ponce, M. Hebert, Schmid. C., and A. Zisserman, Eds. Springer, 2006, ch. What and where: 3D object recognition with accurate pose, pp. 67–82.

[2] M. Özuysal, P. Fua, and V. Lepetit, "Fast Keypoint Recognition in Ten Lines of Code," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[3] A. Collet, D. Berenson, S. Srinivasa, and D. Ferguson, "Object recognition and full pose registration from a single image for robotic manipulation," in *IEEE International Conference on Robotics and Automation (ICRA '09)*, May 2009.

[4] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[6] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce, "3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints," *Int. J. Comput. Vision*, vol. 66, pp. 231–259, March 2006.

[7] J. Sivic and A. Zisserman, "Efficient visual search of videos cast as text retrieval," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 4, pp. 591 –606, 2009.

[8] R. Fergus, P. Perona, and A. Zisserman, "Weakly supervised scale-invariant learning of models for visual recognition," *Int. J. Comput. Vision*, vol. 71, pp. 273–303, March 2007.

[9] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *Int. J. Comput. Vision*, vol. 77, no. 1-3, pp. 259–289, 2008.

[10] L. Vacchetti, V. Lepetit, and P. Fua, "Stable Real-Time 3D Tracking using Online and Offline Information," *PAMI*, 2004.

[11] M. Özuysal, V. Lepetit, F. Fleuret, and P. Fua, "Feature Harvesting for Tracking-by-Detection," in *European Conference on Computer Vision*, vol. 3953, 2006, pp. 592–605.

[12] M. Grabner, H. Grabner, and H. Bischof, "Learning Features for Tracking / Tracking via Discriminative Online Learning of Local Features," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, 2007.

[13] P. M. Roth, M. Donoser, and H. Bischof, "On-line Learning of Unknown Hand Held Objects via Tracking," in *Proc. 2nd International Cognitive Vision Workshop (ICVS)*, Graz, Austria, 2006.

[14] H. Riemenschneider, M. Donoser, and H. Bischof, "Robust Online Object Learning and Recognition by MSER Tracking," in *Proc. 13th Computer Vision Winter Workshop (CVWW)*, 2007.

[15] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 2006.

[16] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[17] Q. Pan, G. Reitmayr, and T. Drummond, "ProFORMA: Probabilistic Feature-based On-line Rapid Model Acquisition," in *Proc. British Machine Vision Conference (BMVC)*, 2009.

[18] T. Weise, T. Wismer, B. Leibe, and L. V. Gool, "Online loop closure for real-time interactive 3d scanning," *Computer Vision and Image Understanding*, vol. 115, no. 5, pp. 635–648, 2011, special issue on 3D Imaging and Modelling.

[19] S. Ekvall, D. Kragic, and F. Hoffmann, "Object recognition and pose estimation using color cooccurrence histograms and geometric modeling," *Image and Vision Computing*, vol. 23, no. 11, pp. 943–955, 2005.

[20] M. M. Torres, A. C. Romea, and S. Srinivasa, "Moped: A scalable and low latency object recognition and pose estimation system," in *Proceedings of ICRA 2010*, May 2010.

[21] A. C. Romea and S. Srinivasa, "Efficient multi-view object recognition and full pose estimation," in *2010 IEEE International Conference on Robotics and Automation (ICRA 2010)*, May 2010.

[22] S. Hinterstoisser, S. Benhimane, and N. Navab, "N3m: Natural 3d markers for real-time object detection and pose estimation," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, Oct. 2007, pp. 1 –7.

[23] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *J. ACM*, vol. 45, pp. 891–923, November 1998. [Online]. Available: http://doi.acm.org/10.1145/293347.293348

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

[25] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[26] B. M. Haralick, C.-N. Lee, K. Ottenberg, and M. Nlle, "Review and analysis of solutions of the three point perspective pose estimation problem," *International Journal of Computer Vision*, vol. 13, pp. 331–356, 1994.

[27] M. A. Lourakis and A. Argyros, "Sba: A software package for generic sparse bundle adjustment," *ACM Trans. Math. Software*, vol. 36, no. 1, pp. 1–30, 2009.

[28] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, pp. 603–619, May 2002.

[29] C. Laporte and T. Arbel, "Efficient Discriminant Viewpoint Selection for Active Bayesian Recognition," *International Journal of Computer Vision*, vol. 68, no. 3, pp. 267–287, 2006.

[30] S. Helmer, D. Meger, M. Muja, J. J. Little, and D. G. Lowe, "Multiple Viewpoint Recognition and Localization," in *Proceedings of the Asian Computer Vision Conference*, Queenstown, New Zealand, 2010.

[31] D. Meger, A. Gupta, and J. Little, "Viewpoint detection models for sequential embodied object category recognition," in *IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, AK, 2010, pp. 5055–5061.

[32] M. Kopicki, R. Stolkin, S. Zurek, T. Mörwald, and J. Wyatt, "Predicting workpiece motions under pushing manipulations using the principle of minimum energy," in *Proceedings of the RSS workshop*, 2009.