Recognition self-awareness for active object recognition on depth images

Andrea Roberti
andrea.roberti@univr.it
Marco Carletti
marco.carletti@univr.it
Francesco Setti
francesco.setti@univr.it
Umberto Castellani
umberto.castellani@univr.it
Paolo Fiorini
paolo.fiorini@univr.it
Marco Cristani
marco.cristani@univr.it

Department of Computer Science
University of Verona
Verona, Italy

Abstract

We propose an active object recognition framework that introduces the *recognition self-awareness*, which is an intermediate level of reasoning to decide which views to cover during the object exploration. This is built first by learning a multi-view deep 3D object classifier; subsequently, a 3D dense saliency volume is generated by fusing together single-view visualization maps, these latter obtained by computing the gradient map of the class label on different image planes. The saliency volume indicates which object parts the classifier considers more important for deciding a class. Finally, the volume is injected in the observation model of a Partially Observable Markov Decision Process (POMDP). In practice, the robot decides which views to cover, depending on the expected ability of the classifier to discriminate an object class by observing a specific part. For example, the robot will look for the engine to discriminate between a bicycle and a motorbike, since the classifier has found that part as highly discriminative. Experiments are carried out on depth images with both simulated and real data, showing that our framework predicts the object class with higher accuracy and lower energy consumption than a set of alternatives.

1 Introduction

Active object recognition (AOR) allows to consider different views of the test object, overcoming the single-view hypothesis of classical object recognition, making the classification problem much easier in principle. Unfortunately, this freedom comes with a price, which is that of maneuvering the sensor for selecting informative images: in fact, not all the views are equally discriminative [21], too similar images are not informative and the movements of the camera are usually bounded by manipulability and energy constraints [11]. Active object recognition means actually to long-term planning in order to reach a good trade-off between a higher classification accuracy and a lower cost for moving the sensor.

Many AOR approaches have been designed, with the POMDP-based techniques being the most effective ones [1, 7, 14]. Strictly speaking, all of them treat the classifier as a black box, maximizing entropy principles among the acquired images [7], possibly focusing on geometric saliency cues of the test object [1].

In this paper, we propose the first AOR approach which does open the box of the classifier, understanding its capabilities, injecting thus a *recognition self-awareness* in the planning process. Technically, we model the camera planning as a Partially Observable Markov Decision Process (POMDP), in line with the most successful non-myopic techniques in the literature [1, 7, 14]. In particular we design a novel observation model (that is, how the planning process manipulates the information gathered from the scene) that exploits deep network visualization techniques [6, 23]. Deep visualization approaches provide *qualitative* explanations on why a particular classifier succeeds or fails in classifying 2D images. In particular, we are interested in those approaches which provide saliency maps over the input images [6]. Here we adopt such saliency maps in a *quantitative* fashion, by building a 3D dense saliency volume which fuses together saliency maps obtained from different viewpoints, obtaining a continuous proxy on which parts of an object are more discriminative for a given classifier. This volume is injected in the POMDP observation model; in this way, the robot can move knowing the capabilities of its classifier. We call our model *Recognition Aware*-POMDP (RA-POMDP). As an example, if the robot knows that its classifier is effective in distinguishing between a motorbike and a bike by looking for the presence of an engine, it will move the camera in a side view (where the engine can be easily spotted) instead of focusing on frontal views (which intuitively may also work, but not for that specific classifier).

Experiments have been carried out on ObjectNet3D [22], a dataset of 3D models of objects belonging to 85 semantic classes, exploiting as sensor a depth camera, with simulated and real robots, showing dramatic improvement against state-of-the-art approaches. Ablation studies show the effectiveness of our RA-POMDP against alternative models.

The rest of the paper is organized as follows: in Sec. 2 we briefly review the recent AOR literature and the deep network visualization; the problem formalization and our RA-POMDP are in Sec. 3 and 4, respectively; finally, an extensive experimental campaign is reported in Sec. 5.

2 Related Work

Active object recognition. Several works on AOR focus on selecting the next best view within a finite set of candidates [4, 8, 15]. In [15], a belief model of the unobserved space is exploited to estimate the expected information gain of each possible viewpoint. Similarly, in [4], the next-best-view prediction is based on Hough Forests running on unsupervised features learned from depth-invariant patches using a sparse autoencoder. In [8], a greedy approach is used to maximize the conditional entropy of the next view. All these methods are myopic, only considering the short-term (next-time step) reward. Long-term planning is modeled with reinforcement learning, in order to reach a good trade-off between a higher classification accuracy and a lower cost for moving the robot. Foundational work [13] plans an optimal sequence of views that maximally discriminates objects of different classes and their orientations. In [7], a probabilistic model is used to encode structural relations among objects and locations. An object search task is then represented by fitting the probabilistic model with the visual appearance of the object of interest. A sequence of views is planned using a POMDP with conditional entropy as the reward function. The approach of [1] is a non-myopic strategy formulated as an active hypothesis testing problem solved with a point-based approximate POMDP algorithm; it uses 3D point clouds as input data for a viewpoint pose tree classifier and dynamic programming to solve an infinite-horizon planning problem. As an extension of this work, [17] includes an energetic term in the cost function to be optimized. In a very recent work [14], a variation of Monte Carlo tree search is employed to achieve a non-myopic planning. A particle filter is combined with Gaussian process regression to estimate joint distributions of object class and pose, and predict sensor observations from future viewpoints. Differently from the other POMDP-based methods, this work exploits Monte Carlo methods to avoid full-width expansion in the search space.

POMDP-based approaches suffer from two main problems: 1) intractability in the continuous search space, and 2) the observation model is usually computationally expensive (see Sec. 4.1). In this work we build upon a point-based algorithm to compute approximate solutions [10], by considering the 3D dense saliency volume.

Visualization of deep networks. The works of [3, 23, 24, 26] individuate those images which activate a certain neuron the most. Other approaches consider the network as a whole, generating dreamlike images that bring the classifier to high classification scores [19, 24]. Another type of deep visualization highlights those salient patterns which drive a classifier toward a class [5, 6, 12, 18, 25, 27] or against it [28].

In all of these cases, the output of a visualization approach is merely used to unveil hidden patterns the network is more sensible to, and it is constrained to 2D images. Here we build a 3D volume using the visualization results over different views and we use it in a quantitative way, embedding it in the observation model of a POMDP.

3 Problem formulation

We consider a multi-class classification scenario, where the class labels belong to the finite set $\mathcal{C}$. A single instance of a given class is a 3D object which is located with a standard pose in the centroid of a spherical workspace V_ρ of radius ρ where a robotic arm can move in the upper hemisphere (since the object lies on a solid floor). The centroid of the object coincides with the centroid of V_ρ. A depth sensor is mounted on the end-effector of the robotic arm[1]. The robot can move and acquire a depth image at each time step, until it stops and provides the object class. The decision whether to move or to stay, and in case where to go, is taken by minimizing an energy function that combines the cost for moving the robot and acquire a new view, E_M, and the cost for an incorrect classification, E_c:

$$E = E_M(x,x') + \lambda\, E_C(c,\hat{c}) \tag{1}$$

where c and $\hat{c}$ are the predicted and correct classes respectively, x and x' are two generic locations in the 3D space, and λ is a constant value. The process iterates over time and it stops when the cost for moving the robot becomes higher than the cost associated to the

[1] We use a depth sensor to simplify the creation of the 3D dense saliency volume. In any case, RGB-D data can be also considered, and will be the subject of future work.

classification error. In this work, we define the classification cost as a constant value that uniformly penalizes incorrect classification:

$$E_C(c,\hat{c}) = \begin{cases} 0 & c = \hat{c} \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

while the movement cost takes into account two different terms: one related to geometrical properties, *i.e.* the length of the trajectory from x to x', and the second related to the kinematics of the robot itself:

$$E_M(x,x') = len(x,x') + man(q') \tag{3}$$

Here q' is the configuration of the robot's joints when the sensor is positioned at x', and $man(q')$ represents the *manipulability measure*, *i.e.* an estimate of the capacity of change in position and orientation of the robot end-effector given a joint configuration. Formally this measure is defined as $man(q) = \sqrt{\det\left(J(q)\, J^T(q)\right)} \geq 0$ where $J(\cdot)$ is the Jacobian operator, and $man(q) = 0$ coincides with a singular configuration. No other active object recognition approach considers the manipulability measure. To minimize Eq. 3, we consider the POMDP framework, explained in the following section.

4 Our model RA-POMDP

We restrict the moving sensor to stop and look towards the centroid of the sphere V_ρ at a finite set of viewpoints $\mathbf{x}_i \in \mathcal{X}_\rho \subset V_\rho$: this allows us to ensure that the sensor is always pointing to the object of interest. From now on, the robot iterates over the following steps: 1) decide if moving to a new position is convenient (otherwise the class label has to be provided and the process stops), 2) retrieve the best move from the POMDP policy, 3) move and acquire a new image, 4) update the belief state of the POMDP. The process starts with the robot in a random but known position.

4.1 Partially Observable Markov Decision Process

A POMDP is a 6-tuple (S,A,T,R,Ω,O), where S is a finite set of states, A is a finite set of actions, $T : S \times A \rightarrow S$ is the transition function defining the probability of state change upon application of a given action, $R : S \times A \rightarrow \mathbb{R}$ is the reward function that represents the reward granted to the system after having reached the new state with the given action, Ω is a finite set of observations, and O is the probability distribution of the observations according to the states and the actions.

At each time step, given a current state $s \in S$, the agent receives an observation $o \in \Omega$ with probability $O(s,o) = Pr(o \mid s)$. Depending on this observation and the current state, the agent takes an action $a \in A$, which causes a transition to state s' with probability $T(s,a,s') = Pr(s' \mid s,a)$. Finally, the agent receives a reward r equal to $R(s,a)$. Then the process repeats.

In our RA-POMDP formulation, the state s at time t is a pair $\langle \mathbf{x}_t, c \rangle$, where $\mathbf{x}_t \in \mathcal{X}_\rho$ is the viewpoint (here assumed to be measurable), and $c \in \mathcal{C}$ is the (hidden) class of the object in the scene. In other words, we assume the robot as knowing its position at each time step[2]. This assumption is reasonable since the motion is deterministic (we have the set $\mathcal{X}_\rho$ of

[2] Relaxing this constraint means that the robot, after each movement, has to check its position w.r.t. a reference system. In this work we ignore this aspect which could be considered as future work.

finite viewpoints), thus the set of actions is to move between viewpoints, and the transition function only affects the viewpoint part of the state (the object class label does not change) and has the form:

$$T(s,a,s') = \begin{cases} 1 & a \text{ is “move from } s \text{ to } s'\text{”} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

The observation o corresponds the output $\mathbf{z}_t$ of the static classifier, while the observation model $O(s,o)$ is generated at training time as described in the next section.

Solving a MDP means to find an optimal policy mapping from a state to an action that maximizes the expected total reward. However, since in a POMDP the state is partially observable, the concept of *belief* has to be taken into account. A belief is a probability distribution over all the states $s \in S$. A POMDP policy π maps a belief b to a prescribed action a. A policy π induces a value function $V_\pi(b)$ that specifies the expected total reward of executing the policy π starting from b. The goal for the robot is to choose the optimal policy π^*, *i.e.* the policy that maximizes the associated value function: $V^* = \mathbb{E}\left[\sum_t r_t\right]$. This problem is usually computationally intractable, but approximate solutions have been proposed in the literature. In this work we use SARSOP approach [10] that finds the best policy iteratively by sampling points in the belief space and pruning away the non optimal candidates. Starting from an initial distribution b_0, at every iteration the belief is updated using the formula:

$$b'(s') = \alpha\, O(s',o) \sum_{s \in \mathcal{S}} T(s,a,s')\, b(s) \tag{5}$$

where α is a normalization constant and all the new beliefs are guaranteed to be reachable from b_0. In this setup, we introduce a novel observation model $O(s',o)$.

4.2 RA-POMDP observation model: 3D dense saliency volume

Our RA-POMDP supposes that a 3D object classifier is trained before to operate on the robot. At training time, we generate a set of synthetic depth images $\mathcal{D}$ by projecting artificial 3D models on a simulated depth camera located in a set of viewpoints uniformly distributed on the surface of a sphere centered at the centroid of each model (Fig. 1a). Note that, differently from [1], with our approach these training viewpoints are conceptually unrelated to the viewpoints $\mathcal{X}_\rho$ used at testing time, as well as to the radius of the viewsphere V_ρ.

The output of the classifier is a distribution $\mathbf{z}$ that returns the probability of the object to belong to each class in $\mathcal{C}$: $\mathbf{z} = [z_1 \ldots z_{|\mathcal{C}|}]$, $\sum_{c=1}^{|\mathcal{C}|} z_c = 1$.

Our goal is to understand how the classifier uses the input to classify, *i.e.* which depth image regions have been considered more important to decide a particular class. First, for each viewpoint specific depth image $\mathcal{D}_\mathbf{x}$ we compute a 2D visualization map ($\mathcal{S}^2_\mathbf{x}$) as in [6]. We do so by learning a mask whose perturbation (usually by blurring its corresponding pixels) drifts the classifier away from deciding the correct class, causing a drop in the related entry of $\mathbf{z}$ (Fig. 1b). The mask is dense and each of its pixel intensities is proportional to the classification drop occurred when masking it. The 2D mask is then mapped to the 3D volume by means of the associated depth map. The 3D model is now discretized into a finite set of voxels, and to each voxel we associate a 3D saliency score $\mathcal{S}^3(v)$ computed as the median value of the saliency of all the points lying inside it (Fig. 1c).

We are now ready to define the observation model O for the POMDP planner. We assign to each viewpoint a cumulative score (Fig. 1d) by averaging the 3D saliency value of all the

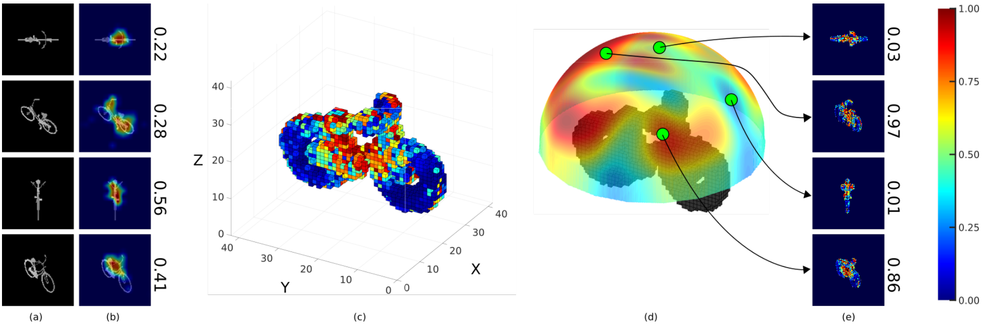

Figure 1: Overview of the RA-POMDP observation model construction: (a) depth maps; (b) saliency maps of the deep classifier; (c) 3D volume of the object; (d) observation model with (e) saliency maps on generic views, whose pixel summation is then mapped on the hemisphere.

voxels in the field of view of the camera (Fig. 1e). This viewpoint specific score is then normalized in order to guarantee that for each class the observation model is in the range $[0,1]$. Mathematically, the observation model returns an estimate of the classifier output $\mathbf{z}$ as a function of the measured part of the state ($s = \mathbf{x}$), and, given the transition function of Eq. 4, we can express it as a function of the next state:

$$O(s',o) = Pr(\mathbf{z} \mid s') = \alpha \sum_{i=1}^{N_v} \mathcal{S}^3(v_i)\, \eta(v_i, \mathbf{x}') \tag{6}$$

where α is a normalization factor, N_v is the total number of voxels, and $\eta(v_i, \mathbf{x}')$ is a visibility function that returns 1 if the i-th voxel v_i is visible from $\mathbf{x}'$, and 0 otherwise.

5 Experiments

We evaluate two main aspects of our RA-POMDP: 1) the quality of the 3D saliency volume (Sec. 5.1), and 2) the recognition performances, on both simulated and real data (Sec. 5.2).

5.1 3D saliency volume

We present some results on different object classes, showing how the saliency volume changes in dependence with the type and number of classes of 3D objects considered.

As dataset, we consider the ObjectNet3D [22] dataset, commonly adopted for passive 3D object recognition. The dataset is composed of RGB images and CAD models, and we use the latter for the experiments. In particular, we select three different sets of classes, composed respectively by 2, 35 and 85 classes (the whole dataset), considering the same number of 3D models for each class in order to avoid biases in the 3D saliency volume. Specifically, we pick 5 random models from each class (the minimum number of models for a class): 3 models for training, 1 for building the 3D dense volume and 1 for the testing. Experimentally, fusing together different models of a single class in for creating the 3D dense volume was not found as particularly beneficial for the active recognition stage; in any case, we plan to further investigate this aspect as a next step.

For each 3D model, we extract depth acquisitions with the V-REP software[3], in particular simulating the real camera (Asus Xtion Pro) that has been used for the real data experiments. With V-rep, we define the set $\mathcal{X}_\rho$ of 128 views, uniformly distributed on the upper hemisphere of radius 0.6 meter.

5.1.1 3D saliency volume creation

The steps of the 3D saliency volume have been already shown in Fig. 1, focusing on the `bicycle` class. Once we have computed the 2D saliency of each view using [6], we build the 3D model of the target class using depth images and averaging the saliency of the views that insist on the same voxel. As for 3D classifier, we adapt an ImageNet-pretrained AlexNet architecture [9] similarly to [2]. Specifically, we substitute the three fully connected layers with a single one, mapping the convolutional features directly to the desired number of classes. For fine-tuning the classifier, we employed the ADAM optimizer with a learning rate of 0.0001 and weight decay 0.0005, batch size of 128, for a total of 4 epochs. The proposed set of hyperparameters is enough to reach a single image testing accuracy of 46% over the set of 35 classes, 36% over the set of 85 classes.

For the 3D saliency volume creation, the size of the voxels is not really crucial and goes in the range from 25 to 80. In practice, the voxelization ensures a sort of classification generalization to the other models of the class: voxels smaller than 25 lead to overfitting (bringing to low classification accuracy), higher than 80 lead to a less effective 3D dense volume.

5.1.2 Impact of the type of object classes

The saliency volume is the proxy of what the classifier has found as particular discriminative in a multi-class classification setup. It is thus interesting to observe the impact that different object classes have on the volume of a given class.

To this sake we consider a two-class classification problem, keeping one class fixed, `bicycle`, while changing the other class as `teapot`, `glasses`, and `motorbike`, respectively. The three cases are reported in Fig. 2. As visible, in the teapot case (Fig. 2a) the bike saliency volume is uniformly highlighted: every view serves to discriminate against the teapot. In the case of the glasses (Fig. 2b), the shape and relative location of the lenses resembles the shape of the tires of the bike (note that in the dataset bike and glasses have the same dimensions). As consequence the bicycle tires have less importance than the previous case. In the third case, bike and motorbike are compared (Fig. 2c). Here the tires have definitely less importance, while the internal framework becomes crucial (in practice, the classifier has understood the presence or absence of the engine as discriminative).

5.1.3 Impact of the number of object classes

Another important aspect is to check how the saliency volume changes while increasing the number of classes into play. The question is whether a bigger number of classes would lead to have saliency volume focusing on fewer parts. We focus on the bike class and start with two classes (bike and teapot). The saliency volume of the bike is the same than Fig. 2c. With 35 classes (Fig. 2d) the general aspect of the volume does not change, and with with 85 classes

[3] `http://www.coppeliarobotics.com/`.

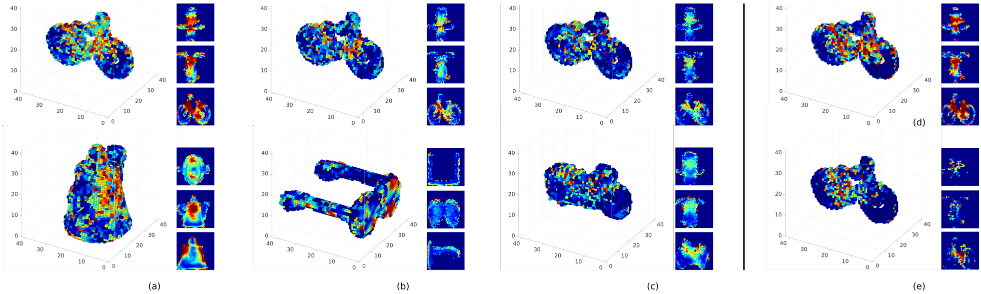

Figure 2: On the left, 3D dense saliency volumes when changing type of classes in a two class problem: bike VS teapot (a), bike VS glasses (b) and bike VS motorbike (c). On the right, when increasing the number of classes: 35 classes (d) and 85 classes (e).

(Fig. 2e) the most important part remain the framework, with the tires that become quite irrelevant, due also to the presence of other classes having the tires (`car`, `wheelchair`).

5.2 Recognition results

We show here the recognition performances of our RA-POMDP, comparing against four competitors, on simulated and real data:

Static. The standard (passive) recognition baseline approach: we take a single observation from the starting viewpoint and predict the class.

Random. A random walk on the viewsphere, avoiding to revisit viewpoints. At each step, the next viewpoint to visit is selected at random among the 8 nearest neighbours.

VP-tree [1]. Like ours, this method is based on a non-myopic POMDP implementation, but in this case the classifier (a vocabulary tree) is treated as a black box, with no saliency computation therein.

Classifier. In this case we change our RA-POMDP model by closing the box of the classifier (the deep network), that is, without any saliency assumption.

As robotic platform, we use a Panda arm, from Franka Emika GmbH[4]. This is a 7 d.o.f. manipulator that can move in a workspace of about 855mm, perfectly suited for our scenario. The motion planner is implemented in the *MoveIt!* framework, part of the Robotic Operating System (ROS) [16]. We use OMPL (Open Motion Planning Library) [20] as motion planner library and *Trac-IK* as kinematic solver. In the simulations, we simulate the RGB-D camera acquisition with *V-Rep*, which has set to emulate an Asus Xtion Pro Live[5].

5.2.1 Simulated data

Simulated data exclude all the variability related to the depth sensor acquisition (sensor noise mainly). For time reasons we are able to test 35 classes out of 85 (the list is in the additional material), repeating the experiment for each class 5 times by sampling randomly the initial position over the 64 $\mathbf{x}_i$ points $\in \mathcal{X}_\rho$.

Table 1 shows the results of the simulations considering the average classification accuracy (*Accuracy*). The other considered quantities are the *Belief* (the belief entry $b(s)$ of the

[4] `https://www.franka.de/panda`

[5] `http://xtionprolive.com/asus-3d-depth-camera/asus-xtion-pro-live`

Approach	Accuracy	Belief	# steps	Distance	Cost
Static	0.36	0.344	–	–	–
Random	0.85	0.584	3.938	0.492	7.837
VP-tree [■]	0.40	0.318	13.350	1.677	10.253
Classifier	0.65	0.567	3.879	0.788	8.005
RA-POMDP	1.00	0.771	4.688	0.814	6.498

Table 1: Quantitative results on simulated data. Average values on objects belonging to 35 classes and 5 initial positions for each class.

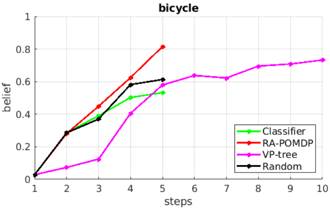

Figure 3: Belief evolution for class bicycle during the exploration.

output class s in the belief distribution b, see Eq. 5), the *Number of steps* (the number of atomic movements performed by the robot), the *Distance* (geodesic distance covered by the camera); the *Energy* (energy spent by the robot as in Eq. 3). These numbers should be considered together with Fig. 3, showing the process of exploration of the robot in terms of steps done and updated belief (in this case, the bicycle class has been reported as representative of the various approaches, for the sake of visualization[6]). As visible, RA-POMDP achieves the highest accuracy and the lowest energy spent. Fig. 3 shows that RA-POMDP has the steepest gradient, meaning that at each step the belief has a considerable improvement. The Classifier approach shows the importance of opening the box of the classifier: in practice, in this approach the observation model has a scattered volume of classification values to consider as proxy, which do not communicate which part has been actually important to be visualized for a successful classification, resulting in an extremely discontinuous observation model: the low Belief in fact says that at some point the next useful point to move is too far and expensive and the process stops. Fig. 3 shows a flatter gradient. VP-tree has even lower performances, dictated by the scarcer classifier taken into account, and (Fig. 3) a longer process of belief update, due to the fact that at each step the classifier is not so discriminative. Please consider that [■] represents the actual state of the art as for active object classification in a very similar scenario (despite it has been proven with less classes, with another dataset). The random classifier has remarkable accuracy performance and cost, but the low belief value means that it has been arrived in a configuration where the next move costs too much than the expected belief positive update: this is due to the fact that random positions do not take into account of the energy and manipulability constraints. Random is better than VP-tree because of a better classifier. Fig. 3 for the random classifier shows an obviously irregular belief update.

5.2.2 Real data

We also tested our framework on a real setup. An RGB-D sensor Asus Xtion Pro Live has been mounted on the end effector of a Franka Panda robotic arm and the exploration has been carried out on 4 object classes. Due to the setup constrains, we focused on objects that can reasonably lie on a tabletop: `cup`, `eyeglasses`, `bottle`, and `scissors`. Results are comparable with the ones in the simulated scenario, with RA-POMDP achieving the highest accuracy, followed by Classifier and Random walks. RA-POMDP is in general able to generate a correct prediction sooner than most of the competitors, spending less energy during the exploration and with a higher confidence (belief). A demo video for the `cup`

[6] doing an average over the classes here would mean to average trajectories of different lengths, resulting in a confused visualization.

scenario is provided in the supplementary material.

6 Conclusions

Active object recognition approaches traditionally considered the classifiers as black boxes, planning robot trajectories to feed them with maximally different images. With RA-POMDP we drastically change this point of view, considering that classifiers build internal representations in which some visual patterns are more important than others. Leveraging deep visualization approaches, by means of a 3D dense saliency volume, we extract this knowledge, exploiting it to plan maximally effective sensor trajectories. Results promote our idea, still at its infancy, as a promising approach with the possibility to easily embed new features.

Acknowledgement

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 779813 (SARAS project). This work has been partially supported by the project of the Italian Ministry of Education, Universities and Research (MIUR) "Dipartimenti di Eccellenza 2018-2022".

References

[1] Nikolay Atanasov, Bharath Sankaran, Jerome Le Ny, George J. Pappas, and Kostas Daniilidis. Nonmyopic view planning for active object classification and pose estimation. *IEEE Transactions on Robotics*, 30(5):1078–1090, 2014.

[2] Fabio Maria Carlucci, Paolo Russo, and Barbara Caputo. A deep representation for depth images from synthetic data. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[3] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] Andreas Doumanoglou, Rigas Kouskouridas, Sotiris Malassiotis, and Tae-Kyun Kim. Recovering 6D object pose and predicting next-best-view in the crowd. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341:3, 2009.

[6] Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[7] Marc Hanheide, Charles Gretton, Richard Dearden, Nick Hawes, Jeremy Wyatt, Andrzej Pronobis, Alper Aydemir, Moritz Göbelbecker, and Hendrik Zender. Exploiting probabilistic knowledge under uncertain sensing for efficient robot behaviour. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

[8] Vasiliy Karasev, Alessandro Chiuso, and Stefano Soatto. Controlled recognition bounds for visual learning and exploration. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[10] Hanna Kurniawati, David Hsu, and Wee Sun Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and systems*, 2008.

[11] Jean-Claude Latombe. *Robot motion planning*.

[12] Aravindh Mahendran and Andrea Vedaldi. Salient deconvolutional networks. In *European Conference on Computer Vision (ECCV)*, 2016.

[13] Lucas Paletta and Axel Pinz. Active object recognition by view integration and reinforcement learning. *Robotics and Autonomous Systems*, 31(1-2):71–86, 2000.

[14] Timothy Patten, Wolfram Martens, and Robert Fitch. Monte Carlo planning for active object classification. *Autonomous Robots*, 42(2):391–421, 2018.

[15] Christian Potthast and Gaurav S. Sukhatme. A probabilistic framework for next best view estimation in a cluttered environment. *Journal of Visual Communication and Image Representation*, 25(1):148–164, 2014.

[16] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. ROS: an open-source robot operating system. In *ICRA workshop on Open Source Software*, 2009.

[17] Andrea Roberti, Riccardo Muradore, Paolo Fiorini, Marco Cristani, and Francesco Setti. An energy saving approach to active object recognition and localization. In *Annual Conference of the IEEE Industrial Electronic Society (IECON)*, 2018.

[18] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[19] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[20] Ioan A Sucan, Mark Moll, and Lydia E Kavraki. The open motion planning library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, 2012.

[21] Dong Wang, Bin Wang, Sicheng Zhao, Hongxun Yao, and Hong Liu. View-based 3D object retrieval with discriminative views. *Neurocomputing*, 252:58–66, 2017.

[22] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. ObjectNet3D: A large scale database for 3d object recognition. In *European Conference Computer Vision (ECCV)*, 2016.

[23] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. In *ICML Workshop on Deep Learning*, 2015.

[24] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference Computer Vision (ECCV)*. Springer, 2014.

[25] Jianming Zhang, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-down neural attention by excitation backprop. In *European Conference on Computer Vision (ECCV)*. Springer, 2016.

[26] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. In *International Conference on Learning Representations (ICLR)*, 2015.

[27] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *IEEE Intrnational Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[28] Luisa M. Zintgracef, Taco S. Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. In *International Conference on Learning Representations (ICLR)*, 2017.