Segmenting Objects through an Autonomous Agnostic Exploration Conducted by a Robot

Leni Le Goff, Ghanim Mukhtar, Pierre-Henri Le Fur, Stephane Doncieux

Segmenting objects through an autonomous agnostic exploration conducted by a robot

Leni K. Le Goff, Ghanim Mukhtar, Pierre-Henri Le Fur and Stephane Doncieux
UMR 7222, ISIR, Sorbonne Universites, UPMC Univ Paris 06, Paris, France
UMR 7222, CNRS, ISIR, Paris, France
Email: {le_goff, doncieux}@isir.upmc.fr

***Abstract*—Human's everyday environment is an open environment in which objects with new shapes, colors or textures frequently appear. Enabling robots to deal with such environments and to manipulate those objects raises a difficult challenge: how to recognize an object ? How to distinguish it from the background ? An approach is proposed here to allow the robot to find this segmentation on its own. It relies on an active exploration of the environment aimed at identifying features of things that move after a contact with robot's end-effector. The only assumption made is that objects of interest are solid objects that the robot can move. The proposed approach can thus be applied without modifications to a large range of environments, as shown by the experiments performed by the robot.**

***Keywords*-Autonomous exploration; Developmental robotics; Interactive Perception; Learning; Computer Vision**

I. Introduction

Our environment is full of objects with diverse size and/or shape. New objects are not rare. Building a robot that could manipulate them without the help of an expert would be a first and significant step towards service robotics. Object manipulation remains a challenge. It requires a fine control of robot arm and end-effector, and also requires objects recognition and segmentation. In this paper, focus is given to objects segmentation. This problem is hard because of the wide variety of objects in a typical human environment. Objects vary by shape, color, size and texture. They also have different functions or physical properties. The environments are also diverse. For instance, in a bathroom, a robot could deal with objects near the bath, the washbasin or closets. This diversity makes it hard to preprogram a robot with the features of all objects and environments it might face.

A robot that makes less assumption about objects or environments features and that could learn on its own to distinguish objects from the background environment would be able to adapt to any situation [1] and could thus face a human environment in all its complexity. This is the main concept of developmental robotics [2], [3], in which the robot must build its own representations of its body and of the world through self-interactions and interactions with the environment. For instance, to build a representation of objects, the robot "plays" with them and extracts the features which characterize them and that are relevant with respect to robot abilities.

Perception driven by manipulation (or opposite) is known as Interactive Perception [4]. By observing the effect of its action on an object, the robot can build a representation in relation

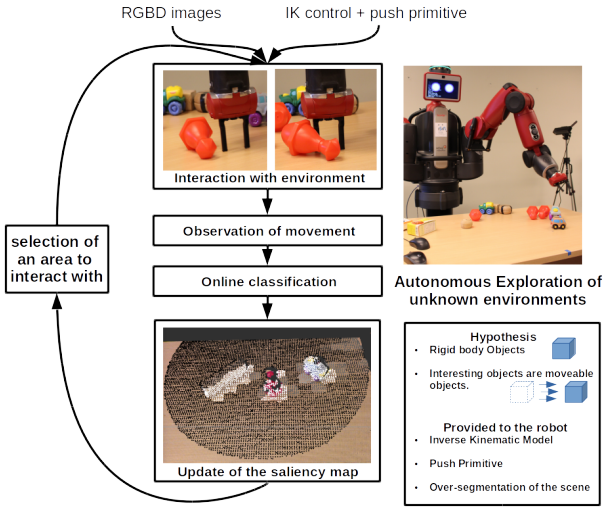


Fig. 1. Overview of the proposed babbling approach. The robot chooses a point of the environment to interact with. It observes if something has moved as a result of its action. A classifier is trained online with the informations gathered thanks to these interactions. Finally, a saliency map is computed to guide the exploration.

to its own abilities. This field was first studied by Metta and Fitzpatrick [5], [6], [7]. They developped a method on a real robot to segment objects from a table. The robot, by pushing an object and observing the resulting movement, is able to separate the objects from the background. Interactive Perception is also related to the concept of affordances introduced by J. J. Gibson [8]. This concept highlights that objects have inherent "values" and "meanings" which could be perceived by an agent and could be linked to its possible actions on those objects. Sahin et al. highlight the fact that these affordances are to be acquired by interacting with the environment [9].

The goal of this work is to define a method, that uses Interactive Perception, affordances and exploration, *to learn autonomously how to distinguish manipulable elements in an environment*. In other words, how to segment objects from the background with an autonomous exploration (see Figure 1). We assume that *something moveable by robot's end-effector is potentially an object*. So, the robot learns to identify the relevant features of things that are moveable as a result of its action, i.e. objects that "afford" the "move" action. The "move" action is considered here as the action to be applicable

to an object for it to be interesting from robot's point of view.

Most of previous work, in Interactive Perception [4], use a passive image processing step before any interactions to produce objects hypothesis and bootstrap the system. But this step needs assumptions on the structure of the environment and on shapes and/or colors of objects (see section II-A). In this work, to avoid assumptions that are specific to a particular kind of environment, an over-segmentation of the scene into supervoxels (see section III-B) is used to bootstrap the exploration. Then, a saliency map is built online that represents the distribution of features which afford the "move" action, i.e the potential presence of an object (see section III-C). The main contribution of this work is :

The integration of action, perception and learning into a single process to produce a saliency map which represents the distribution of potential presence of objects, in order to segment them from the background in a cluttered environment with a minimum prior knowledge about environment structure.

This work is not focused on object recognition or objects model learning. It comes at a previous step, for a first identification before a more targeted exploration. With a minimum of a priori knowledge on the environment, this work represents the very first step of a developmental process that would aim at the acquisition of robust and adaptive object manipulation skills. An object will be defined here as a cluster of adjacent supervoxels that afford the "move" action for the robot.

II. Related Works

A. Discovering Objects by Interactive Perception

Interactive Perception is used in many works [4] to learn objects appearances, for objects recognition and manipulation, by autonomous exploration with a humanoid robot. But as this work is focused on objects segmentation, this section focuses on works that use Interactive Perception for this purpose. As written in the introduction, Interactive Perception has been studied first by Fitzpatrick and Metta [5], [6], [7]. In their work, a robot learns objects by interacting with them. It uses the link between its action and the effect of its action to segment the object, and then, collect data about it. Their method is restricted to a plane as background. In this paper, the method works on different environment and on many objects as long as they are rigid solids.

Some works on Interactive Perception, use object candidate or hypothesis generation as first step. Objects candidates are clusters of the visual field of the robot built with passive image processing algorithms. These candidates are rejected or confirmed by the robot action [10], [11], [12], [13], [14]. In those works, candidates generation relies on assumption that objects are on a flat surface. In some of these works, they use also simple shape matching to work with textureless object or highly textured background [12], [13], [14]. Works like [11], [15] use object database to simplify object candidate generation.

Most of those works are efficient and successful in segmenting and tracking objects. But the step of object hypothesis generation needs strong assumptions on the environment and the objects, and their generalization leads to computations of increasing complexity.

The objective here is to learn object features from raw sensor data with a single assumption: objects are rigid solids. The robot uses Interactive Perception to learn what is relevant to distinguish background from objects. By starting from a lower level than previous works discussed in this section, less assumptions are required, thus opening the way to a more adaptive robot behavior [1].

B. Scene Understanding by Affordance Recognition

As discussed in section I, Interactive Perception is linked with learning which features in the environment afford this action. E. Gibson studied how children develop affordances and claimed that learning is "discovering distinctive features and invariant properties of things and events" [16] and "discovering the information that specifies an affordance" [17]. Learning affordances and building a meaningful segmentation for a robot is about learning "regularities" in its sensorimotor[1] domain .

In the work of Ugŭr et al. [18], a wheeled robot explores an environment to learn the "traversability" of obstacles. The robot is confronted to obstacles with different shapes : spheres, cylinders, parallelepiped. The "traversability" of an obstacle depends on its shape and its orientation. By trials, the robot must learn which visual feature affords "traversability". This set-up is far from real conditions but their work demonstrates that learning of scene understanding through affordances on a robot is possible.

The works of Popovic et al. and Krüger et al. use the same approach but relate to grasping skill acquisition [19], [20]. Like in the work of Ugŭr et al. [18], their goal is to associate features to potential grasp. They use Early Cognitive Vision (ECV) [21] for preliminary image processing. ECV is a computer vision framework that extracts features with a stereo camera. The features are edges, contours, textures and surfaces. The robot tries to grasp different objects and associate ECV's features to successful grasp. But, ECV needs textured or complex objects to work properly. Finally, in this work, they focused on objects and did not discuss about separating objects from background.

On the contrary, the method presented in this paper aims at scene understanding by considering the whole environment, like the work of Craye et al. [22], in which a wheeled robot builds a map of presence probability of objects in its visual field, by exploring its environment. Each pixel is associated to a value between 0 and 1, that represents the probability to be part of an object. They use 2D colored images segmented into superpixels (e.g. clusters of pixels) and extract their average color. A feature is an array of colors formed by

[1]Sensorimotor refer to the coupling of the sensor system and the motor system for an agent

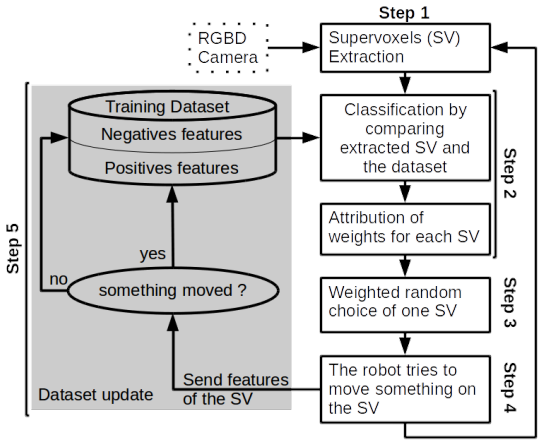


Fig. 2. General workflow of the approach.

averaging the color of a superpixel and the average colors of its neighborhood. Those features can be applicable to all kinds of environments, but, to discriminate objects from background, they also assume that things, on a flat surface, are objects.

By drawing inspiration from these works, the method presented in this paper aims at using Interactive Perception to learn which features afford the "move" action on objects. This work uses as features *average colors* and *normals* of clusters resulting from an over-segmentation. These features are not specific to a certain type of objects.

III. Method

A. Overview

The robot explores an unknown dynamic[2] environment, and this exploration is driven by a saliency map of the environment built online. A saliency map is a distribution of salient parts in the visual field. Salient parts of the visual field are parts which attract gaze [23]. But salience is a subjective concept. If the agent wants to move objects, salient features are those of moveable objects, for instance. This notion is used here: *the robot builds a saliency map representing the distribution of salient features, i.e. features of moveable objects*. Figure 5 shows an example of a saliency map. A RGBD Camera (Microsoft Kinect [3]) is used to retrieve a 3D pointcloud of the scene. This 3D pointcloud is over-segmented into supervoxels thanks to Voxel Cloud Connectivity Segmentation (VCCS) [24] (see section III-B).

Figure 2 presents the general workflow of the method. The exploration is sequential, each iteration is structured into 5 steps :

Step 1 Over-segmentation of the pointcloud into supervoxels. This step is described in section III-B.

Step 2 Computation of the saliency map. A classifier assigns a saliency weight between 0 and 1 to each extracted supervoxel according to the samples stored in the database. These saliences represent the probability for a supervoxel to be part of "something" moveable by the robot, i.e. an object, as we defined it. This step is described in section III-C.

Step 3 Choice of a supervoxel to interact with. The choice is random according to the saliency map, therefore the robot will explore first supervoxels with high saliences.

Step 4 The robot tries to move something on the selected supervoxel. To do it, the robot moves its end-effector towards the center of the chosen supervoxel. Its Inverse Kinematic model is provided for this purpose. Then, when it is around the center of the supervoxel, the robot applies a push primitive in the direction of the center. The push primitive consists in moving its end-effector along a straight line for a fixed distance. Finally, its arm comes back to a position outside of the camera visual field.

Step 5 Observation of a possible effect. The chosen supervoxel is used to create a mask on the 2d colored image to keep just pixels in the area of the supervoxel. Then, images before and after the action of the robot are filtered with the mask and are compared. In this way, differences are checked just around the point of action. Finally, the features are added to the database as a sample labeled positive if there is a difference, negative otherwise (see section III-C).

At the beginning of the exploration all saliences are initialized to 1, because without information about the environment, all the supervoxels are supposed to be interesting and are to be explored. Then, after the first iteration, the random selection is biased by the supervoxels saliences: the probability to explore a supervoxel is its salience normalized by the sum of the saliences of all observed supervoxels:

$$P_{exp}(sv_i) = \frac{s_i}{\sum_j s_j} \quad (1)$$

The definition of those saliencies is described in section III-C.

B. Voxel Cloud Connectivity Segmentation

The saliency map is based on an over-segmentation into supervoxels using Voxel Cloud Connectivity Segmentation [24] (VCCS). As illustrated in Figure 3, supervoxels are clusters of voxels[4].

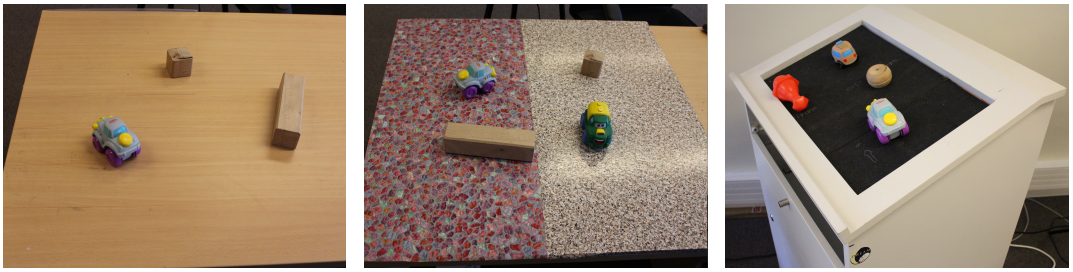
VCCS is similar to superpixel methods, such as SLIC superpixels [25]. It builds clusters of voxels based on colors, normals and shapes features. From seeds evenly distributed on the pointcloud, a region growing method is used to build the supervoxels. The clusters are grown with a local k-means algorithm. Expansion of clusters is controlled by a distance computed with colors, normals and spatial distances. Normals are computed during the algorithm by local plane estimation for each voxel.

The use of 3D information to build supervoxels is a significant enhancement compared to superpixels methods as it allows this segmentation to respect object boundaries. The samples stored to update classification are thus more likely to

[2]In this work, "dynamic" means that the state of the environment is not reinitialized at the begining of each iteration.

[3]Other kind of 3D cameras could be used such as stereoscopic cameras

[4]voxels are the smallest unit of a 3D image like pixels in 2D images



(a) wooden table (b) textured table (c) lectern

Fig. 4. Pictures of the environment used for the experiment : 4a a wooden table, 4b table with textured tablecloth and 4c a lectern.

Fig. 3. Examples of supervoxels segmentation.

be associated to a single object. It removes a significant source of noise in the classification. Also, VCCS works on all kinds of environments because the algorithm uses low level features, such as color, normals and geometric descriptors. Therefore VCCS produces a meaningful over-segmentation of RGB-D images. Moreover, as output, the algorithm provides for each supervoxel its average color and normal, which are used as features for the classification. However a metaparameter (set by the user) controls the size of the supervoxels, so objects must be at least bigger than the size of supervoxels.

C. Building the Saliency Map

When the scene is segmented, each supervoxel is weighted with a value between 0 and 1. These values represent the supervoxels salience. The algorithm, used to compute this salience (see algorithm 1), is a nearest neighbor classifier in the visual features space. The salience of a supervoxel is attributed by comparing its visual features with those stored in the dataset.

At each iteration, all saliences are initialized to 1. Then, according to the dataset, the saliences are increased or decreased with the increment $\gamma*(1-D)$. For each supervoxels and each stored samples, if the label of the current sample is positive (i.e sample from a supervoxel that has moved), $\gamma*(1-D)$ is added to the salience of the current supervoxel and otherwise $\gamma*(1-D)$ is subtracted. Saliences can not be decreased under 0 or increased over 1, to keep them within $[0, 1]$.

Algorithm 1 Algorithm used to update supervoxel saliences. F is a set of samples f stored in the training dataset with their labels (lbl). SVC is a set of supervoxels sv associated with their salience (s), and $feat(sv)$ is the features of sv. ζ and γ are two meta parameters (both are expected to be closer to 0 than to 1). $D \in [0, 1]$

```
procedure UPDATESALIENCES
    for all (lbl,f) in F do
        for all (s,sv) in SVC do
            D = dist(f,feat(sv))
            if D < ζ then
                if lbl = positive then
                    s = s + γ * (1 − D)
                end if
                if lbl = negative then
                    s = s − γ * (1 − D)
                end if
            end if
            if s < 0 then
                s = 0
            end if
            if s > 1 then
                s = 1
            end if
        end for
    end for
end procedure
```

Parameter	ζ	γ	α
Value	0.3	0.05	0.5

TABLE I
VALUES OF THE META-PARAMETERS OF THE CLASSIFIER USED FOR THE EXPERIMENTS

Furthermore, the saliences are updated only if the distance D is lower than a threshold ζ. This threshold represents the radius of influence of a stored sample. A sample will only change the salience of a supervoxel whose features are close to the features of the sample. This limits interferences between samples. For instance, if ζ is equal to 1, a sample

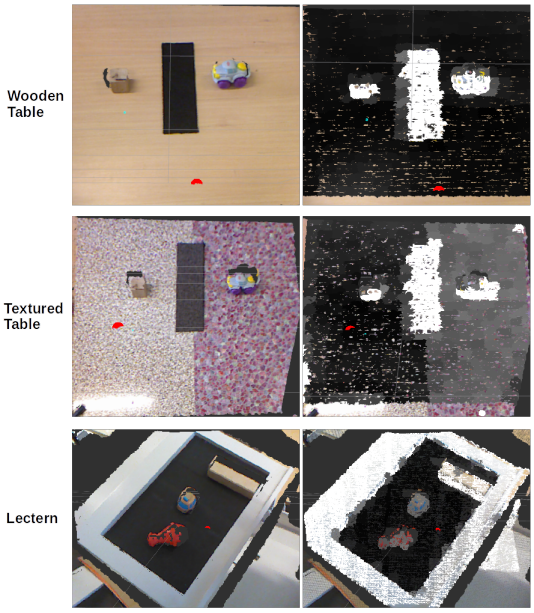


Fig. 5. An example of a saliency map, i.e. distribution of saliences on a set of supervoxels. On the left, a colored pointcloud and on the right, the same point-cloud colored with the corresponding saliences distribution. White corresponds to the most interesting area (supervoxels with a salience of 1) and black to the less interesting area (supervoxels with a salience of 0). These figures have been created with only one sample with the parameter γ fixed at 1 (see section III-C). The red dot indicates the position of the supervoxel selected. The corresponding feature is labelled as non interesting, as a consequence, the features close to it have saliences close to 0. So, the distribution represented on left images is the distance (see equation 2) between the selected supervoxel and the other supervoxels.

will modify the saliences of all observed supervoxels, and so, a single negative sample could decrease the salience of moveable elements. However, if ζ is too low the classifier will need too many samples to learn.

γ is a parameter to control the speed of convergence of the algorithm. If γ is high the saliency will be learned fast but will be very sensitive to false positive or false negative samples. So, γ must be chosen as a compromise between speed and robustness. In the experiments this parameter is fixed at 0.05. In Figure 5, an example of saliency map on each set-up is given. To compute these maps and for illustration, γ is fixed to 1 (its maximum value), so just one sample (with a negative label in this case) is needed to have a saliency map that has converged. Obviously, these saliency maps are not perfect, this can be easily observed on the textured table and the lectern in which the background is not entirely black. Furthermore, any noise or failed attempts will be over-integrated.

D is the distance between the features of two supervoxels:

$$D = \sqrt{\alpha * D_c^2 + (1-\alpha) * D_n^2} \tag{2}$$

D_c is the L_2 distance between average colors of supervoxels and D_n is the L_2 distance between their averages normals; α represents the relative importance of the color with respect to the normal and is between 0 and 1. α is set for the experiments at 0.5, so the color and the normal have an equal influence. Both D_c and D_n are normalized, so, the distance D is between 0 and 1. The values of the parameters used for the experiments described here are given in Table I.

IV. Experiments

A. Set-up

The experiments made to validate the proposed method rely on the Baxter robot. This is a robot with two arms with 7 degrees of freedom each. For the experiment, just one arm is used. The vision sensor is a Microsoft Kinect first version. It provides RGB-D images with a resolution of 640*480.

An experiment is made up with a fixed number of iterations during which the robot tries to reach objects in order to train its classifier, as shown in figure 2. 12 classifiers have been trained like this in 3 different set-ups (4 classifiers each) (see figure 4) : a wooden table, a table with complex textures and a lectern. This last set-up allows to test the method on a non flat environment (e.g. a non tabletop scenario).

B. Evaluation

After training the 12 classifiers, as indicated above, they are evaluated. We propose for the evaluation two metrics:

- *raw performance :* which is an indicator of how well a classifier is able to segment objects from a background identical to the one used during its training.
- *generalization performance :* which is an indicator of how well the classifier is able to segment objects from a background that is different to the one used during its training.

During the training, at each iteration, a snapshot of the classifier is taken, which makes it possible to evaluate its capacity to segment objects at that point of its training. This is done until the end of the training.

Also, for the evaluation, an expert procedure is used to segment objects from background manually[5]. This acts as a reference to count the number of chosen supervoxels that are part of an object. The performances are computed as follows:

1) A naive policy performance to segment objects is estimated, by choosing randomly N supervoxels with an uniform distribution:

$$f_{random} = \frac{n_U}{N} \tag{3}$$

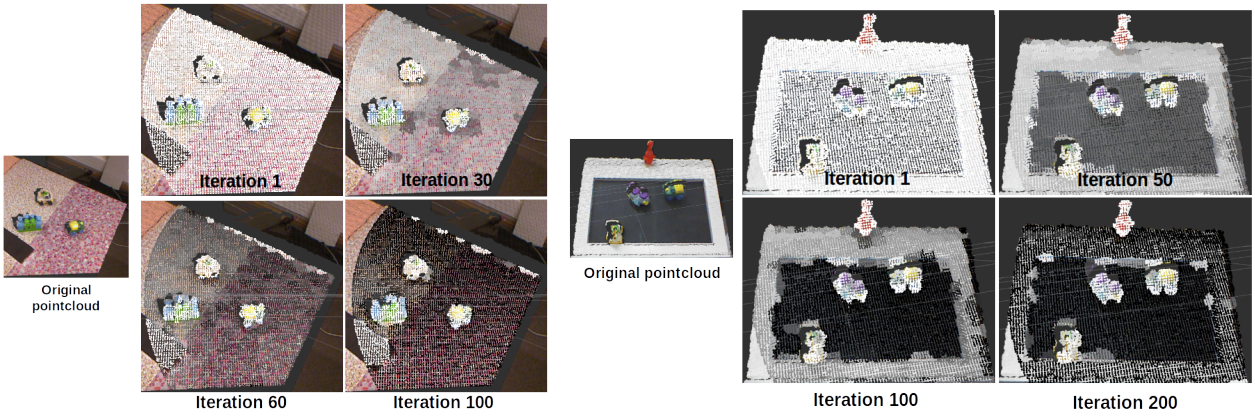
Where: $f_{random} \equiv$ *The naive policy performance index*; $N \equiv$ *Number of supervoxels randomly chosen (N = 100)*; $n_U \equiv$ *Number of supervoxels are actually part of an object.*

2) The classifier snapshot performance is estimated by choosing randomly N supervoxels with a distribution biased by the relevant saliency map (look at equation 1):

$$f_{class} = \frac{n_{SM}}{N} \tag{4}$$

Where: $f_{class} \equiv$ *The classifier snapshot performance*; $N \equiv$ *Number of supervoxels chosen (N = 100)*; $n_{SM} \equiv$ *Number of supervoxels are actually part of an object.*

[5]This manual segmentation is used only for the evaluation, the proposed algorithm does not rely on it.



(a) textured table (b) lectern

Fig. 6. Example of a saliency map during an exploration on the textured table and the lectern. The evolution of the saliency map is depicted at 4 different steps. At iteration 1, as all saliences of the map are initialized at 1, the map is represented in white. Then for the textured table, from iteration 60, the table starts to be black and parts of objects are white.

3) Finally compute an performance score:

$$\eta = \frac{f_{class} - f_{random}}{1 - f_{random}} \tag{5}$$

Where: $\eta \equiv$ *Performance score of the classifier snapshot*; $1 \equiv$ *Expert policy efficiency.*

Thus 1 is the maximum value of efficiency, it is the efficiency expected for an expert controller whereas 0 means that the approach has the same efficiency as that of a naive random controller. The goal of the training process is thus to progressively move from the performance of a naive controller to that of an expert one [26].

For the *generalization performance* of a classifier, classifiers trained on the wooden table are evaluated on the textured table. This case allows to confront the classifier with a more complex environment than the one used during its training. Moreover, those trained on the textured table and on the lectern are evaluated on the wooden table. These two cases allow to confront the classifier with a less complex environment than the one used during its training.

V. Results

Figure 6 shows pictures of the saliency at different moments of the exploration. As explained in section III, the saliency map starts with all saliences at 1, so, all supervoxels are represented in white in the first picture. After several iterations and after having collected new samples, the objects are distinguishable from the background: supervoxels on objects are lighter than those on the background.

The top graphics on Figure 7, represent the raw performance (black curve) and the generalization performance (blue curve) of the classifier. Both performances are computed with equation 5. The bottom graphics represent the accumulated number of negative and positive samples during the exploration of the environment, and the sub-figures 7a, 7b and 7c are respectively the results on the wooden table, the textured table and the lectern.

For each set-up, at a certain iteration (around 40 for wooden table, around 60 for textured table and after 100 for the lectern) the raw performance increases quickly before reaching a plateau. Also, at this moment, the rate of change of both curves, which represents the number of negative and positive samples, changes critically: the first one (negative samples) starts to decline, while the second one (positive samples) increases faster. As represented on the bottom graph of Figure 7, at the beginning, the robot collects almost only negative samples (i.e. interacts only with the background), then at the transition point, its starts to gather more positive samples (i.e. it interacts with moveable elements). Indeed, the absolute difference between the number of positive and negative samples (represented in blue) reaches its maximum around 40 iterations for the wooden table and around 60 iterations for the textured table. At this moment, the classifier has a sufficient number of samples from the background to ”recognize” it. As shown in Figure 6, it corresponds to the moment when the background starts to become entirely dark, and so, the background is classified, with a good certainty, as non moveable elements.

However, the number of negative samples continues to increase. This is due to false negative samples. In some cases, the simple push primitive is not able to move an object. Obviously, a simple push primitive, like used in this work, is not enough to explore fully the environment. This has a little influence on the classifier efficiency because the robot needs to interact several times with the same object to conclude if it is moveable or not. Therefore, this issue only slows down the training.

The classifier needs enough samples about the background to distinguish objects from it, moreover the more complex the background is, the more time is required for the robot to bootstrap its saliency map. Also, on the textured table and the lectern, the efficiency increases more progressively than for the wooden table. Moreover, on the lectern, after 150 iterations, the robot still gathered a lot of samples of the background, even if the raw performance of its classifier has reached a

(a) wooden table (b) textured table (c) lectern

Fig. 7. Top graphics represent the performance (as define in equation 5) over the iterations, in black is represented the performance computed on the same set-up as the one used for training and in blue the performance computed on a different set-up. The second performance represents the generalisation efficiency of the classifier. Bottom graphics represent the accumulated number of positive and negative samples; and the absolute difference between both. Each sub figure is related to different environment : 7a the wooden table, 7b the textured table and 7c the lectern.

plateau. So, for this set-up the robot will need more iterations to fully separate objects from the background.

The raw performance has never reached the maximum score. That is due principally to the lighting variations and the shadows of objects. The classifier could consider a shadow as part of an object. That could be solved by an exploration focused around an object. Furthermore, the training is made with RGB encoding which is very sensitive to lighting variations. This can be fixed by working with HSV encoding which is less sensitive to lighting variations.

For the wooden table, its generalization performance is significantly less than the raw performance, but for the textured table, both performances are close. For the lectern, its generalization performance is significantly better than its raw performance. The wooden table is the simplest set-up in the paper, so, there is less information to collect. Consequently, when the classifier trained on this set-up is evaluated on more complex environments (textured table), information is missing to separate objects from background. On the contrary, the lectern has a lot of visual informations, so the classifier trained on it could be efficient on different and simpler kinds of environments, but only if those environments share visual features with the lectern.

VI. Discussion and Future works

The results show that this online classifier is able to separate objects from the background. It is able to do it in different kinds of environments and with different kinds of objects. For a complex background, the robot must gather more samples than with simple background like an uncolored table. However, even in the hardest set-up (the lectern) tested in this paper, the exploration needs around one hundred iterations to reach a sufficient efficiency. It takes around half an hour for one hundred iterations with our implementation, so, a few hours might be enough to have a performance equal to an expert policy (e.g. hand-made). In future work, longer exploration will be performed to see if the classifier performance can reach the expert one and to check if it over-fits.

Also, the generalization performance, shown on Figure 7, demonstrates that the robot must explore different kinds of environments to collect a wide range of samples. But, even if the classifier seems to have at least some generalization ability, the saliency map produced at the end of a session of exploration is specific to the explored environment. The deceptive aspects of real environments make the generalization difficult. What is learned in a particular situation is not always applicable to another situation. The exploration phase must then be applied during the whole ”life” of the robot to produce a meaningful segmentation of its environment. The classifier presented in this paper, could be used in this way thanks to its speed. Indeed, it learns fast (see section V), and so, it could adapt to new situations and new features quickly. But to avoid over-fitting, a mechanism of unlearning must be used.

The algorithm 1 has a complexity of O(n*m), where n is the size of the dataset and m the number of supervoxels extracted on the current scene. For a large scene, with many supervoxels, and a long exploration, that leads to a large dataset, this algorithm will be very slow. To avoid this problem, a clustering process, such as k-means, or a second learning process, such as neuronal network or random forest, could be applied on the dataset to reduce its size.

This method could be linked to the concept of affordances. The saliency map produced after exploring the environment with a push primitive represents the distribution of visual features that afford a ”push” action. But the features chosen for learning are critical. In this work, the choice was to have features that are neither specific to a kind of environment nor to a kind of object. But, if an object has the same color and shape as a part of the background this will lead to misclassification. This method can be used with other features or descriptors. Also, introducing other modalities such as touch or hearing senses could be interesting.

This link with affordances suggests also to provide more complex capacities of interaction to the robot. The robot could have several ways to interact with objects in order to explore more efficiently its environment.

The experiments presented in section IV show that the hypothesis of *what can be moved by the robot end-effector is an object* is sufficient to produce a meaningful segmentation. This criterion is sufficient to segment objects from background

in the set-ups presented in this paper. Such a segmentation can be used for an exploration focused on objects to collect more accurate and complete information on those objects. Thanks to this dynamic and adaptive segmentation, the robot could explore further how to manipulate objects. This method could be used in a bootstrap phase for the works discussed in section II.

Finally, this method is not limited to objects. It is applicable to all manipulable entities by the robot as long as they are rigid bodies. For instance, the robot could discover manipulable mechanisms, like door handles to turn, buttons on a wall or on a dashboard to push, or a drawer to open. Indeed, the only assumption made on the objects, in this method, is : *all entities that are not part of background are rigid body moveable by the robot independently from the background*.

VII. Conclusion

A method has been proposed to identify the features of objects that can be moved by the robot. This method relies on an exploration of the environment with integration of interactive perception and online learning into a single process. It is aimed at segmenting objects from the background without strong hypothesis on objects or background features. It can thus be applied in different environments without any adaptation, thanks to the saliency map learned by the robot from its own experiences. The results show that the system can indeed extract relevant features and segment objects from the background.

Acknowledgement

This research is sponsored by the DREAM project[6]. This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 640891.

References

[1] S. Doncieux, "Creativity: A driver for research on robotics in open environments," *Intellectica*, vol. 2016/1, no. 65, pp. 205–219, 2016.

[2] M. Asada, K. F. MacDorman, H. Ishiguro, and Y. Kuniyoshi, "Cognitive developmental robotics as a new paradigm for the design of humanoid robots," *Robotics and Autonomous Systems*, vol. 37, no. 2, pp. 185–193, 2001.

[3] A. Cangelosi, M. Schlesinger, and L. B. Smith, *Developmental robotics: From babies to robots*. MIT Press, 2015.

[4] J. Bohg, K. Hausman, B. Sankaran, O. Brock, D. Kragic, S. Schaal, and G. S. Sukhatme, "Interactive perception: Leveraging action in perception and perception in action," *CoRR*, vol. abs/1604.03670, 2016.

[5] G. Metta and P. Fitzpatrick, "Early integration of vision and manipulation," in *Neural Networks, 2003. Proceedings of the International Joint Conference on*, vol. 4. IEEE, 2003, pp. 2703–vol.

[6] P. M. Fitzpatrick and G. Metta, "Towards manipulation-driven vision," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 1. IEEE, 2002, pp. 43–48.

[7] P. Fitzpatrick and G. Metta, "Grounding vision through experimental manipulation." *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, vol. 361, no. 1811, pp. 2165–2185, 2003.

[8] J. J. Gibson, *The ecological approach to visual perception: classic edition*. Psychology Press, 2014.

[9] E. Sahin, M. Cakmak, M. R. Dogar, E. Ugur, and G. Ucoluk, "To Afford or Not to Afford: A New Formalization of Affordances Toward Affordance-Based Robot Control," *Adaptive Behavior*, vol. 15, no. 4, pp. 447–472, 2007.

[10] H. van Hoof, O. Kroemer, and J. Peters, "Probabilistic Segmentation and Targeted Exploration of Objects in Cluttered Environments," *IEEE Transactions on Robotics*, pp. 1–12, 2014.

[11] K. Xu, H. Huang, Y. Shi, H. Li, P. Long, J. Caichen, W. Sun, and B. Chen, "Autoscanning for coupled scene reconstruction and proactive object analysis," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, p. 177, 2015.

[12] D. Schiebener, J. Morimoto, T. Asfour, and A. Ude, "Integrating visual perception and manipulation for autonomous learning of object representations," *Adaptive Behavior*, vol. 21, no. 5, pp. 328–345, 2013.

[13] D. Schiebener, A. Ude, and T. Asfour, "Physical interaction for segmentation of unknown textured and non-textured rigid objects," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 4959–4966, 2014.

[14] K. Hausman, F. Balint-benczedi, D. Pangercic, Z.-c. Marton, R. Ueda, K. Okada, and M. Beetz, "Tracking-based Interactive Segmentation of Textureless Objects," in *Robotics and Automation (ICRA)*, 2013.

[15] N. Bergström, C. H. Ek, M. Björkman, and D. Kragic, "Scene understanding through autonomous interactive perception," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6962 LNCS, pp. 153–162, 2011.

[16] E. J. Gibson, "Perceptual learning in development: Some basic concepts," *Ecological Psychology*, vol. 12, no. 4, pp. 295–302, 2000.

[17] ——, "The world is so full of a number of things: On specification and perceptual learning," *Ecological psychology*, vol. 15, no. 4, pp. 283–287, 2003.

[18] E. Ugur, M. R. Dogar, M. Cakmak, and E. Sahin, "Curiosity-driven learning of traversability affordance on a mobile robot," in *2007 IEEE 6th International Conference on Development and Learning*. IEEE, 2007, pp. 13–18.

[19] M. Popović, G. Kootstra, J. A. Jørgensen, D. Kragic, and N. Krüger, "Grasping unknown objects using an early cognitive vision system for general scene understanding," *IEEE International Conference on Intelligent Robots and Systems*, pp. 987–994, 2011.

[20] N. Krüger, M. Popovic, L. Bodenhagen, D. Kraft, and F. Guerin, "Grasp learning by means of developing sensorimotor schemas and generic world knowledge," in *AISB Convention*. Citeseer, 2011, pp. 23–31.

[21] N. Krüger, N. Pugeault, E. Baseski, L. Jensen, S. Kalkan, D. Kraft, J. Jessen, F. Pilz, A. Kjaer-Nielsen, M. Popovic *et al.*, "Early cognitive vision as a front-end for cognitive systems," in *ECCV 2010 Workshop on Vision for Cognitive Tasks*, 2010.

[22] C. Craye, D. Filliat, and J.-F. Goudou, "Exploration Strategies for Incremental Learning of Object-Based Visual Saliency," *Proc. of the 5th Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics*, pp. 13–18, 2015.

[23] L. Itti and C. Koch, "Computational modelling of visual attention." *Nature reviews. Neuroscience*, vol. 2, no. 3, pp. 194–203, 2001.

[24] J. Papon, A. Abramov, M. Schoeler, and F. Worgotter, "Voxel cloud connectivity segmentation - Supervoxels for point clouds," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2027–2034, 2013.

[25] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC Superpixels," *EPFL Technical Report 149300*, no. June, p. 15, 2010.

[26] S. Doncieux, R. Duro, A. Prieto, F. Bellas, F. Stulp, D. Filliat, T. Hospedales, J. Heinerman, E. Haasdijk, and A. Eiben, "Del. 6.1: Experimental protocol for the validation experiments," DREAM FET H2020, Tech. Rep., 2015.

[6] http://www.robotsthatdream.eu/