Environment Exploration for Object-Based Visual Saliency Learning

Céline Craye[1,2], David Filliat[1] and Jean-François Goudou[2]

Abstract— **Searching for objects in an indoor environment can be drastically improved if a task-specific visual saliency is available. We describe a method to incrementally learn such an object-based visual saliency directly on a robot, using an environment exploration mechanism. We first define saliency based on a geometrical criterion and use this definition to segment salient elements given an attentive but costly and restrictive observation of the environment. These elements are used to train a fast classifier that predicts salient objects given large-scale visual features. In order to get a better and faster learning, we use an exploration strategy based on intrinsic motivation to drive our displacement in order to get relevant observations. Our approach has been tested on a robot in indoor environments as well as on publicly available RGB-D images sequences. We demonstrate that the approach outperforms several state-of-the-art methods in the case of indoor object detection and that the exploration strategy can drastically decrease the time required for learning saliency.**

I. Introduction

Object detection or object search in cluttered environments by mobile robots is still a difficult problem. Despite progress in computer vision for detecting objects in unconstrained images, especially using deep-learning which has recently shown impressive results on complex image datasets such as IMAGENET [21], it remains particularly interesting in a robotics context to move the robot to favorable observation conditions in order to improve recognition performances. Such visual exploration of the environment [3], [8], [17] is often associated with a visual attention strategy so as to direct the robot's attention towards areas of interest, while ignoring irrelevant portions of the visual field.

The selection of areas of interest is typically driven by visual saliency maps [3], [8], [17] or, if depth is available, geometrical segmentation [1], [5]. In the first case, bottom-up saliency maps are based on color images [7], [23] or RGB-D data [20], and highlight stimuli that are intrinsically salient in their context. As interesting elements are not always intrinsically salient, some approach suggest to add top-down modulation in order to further enhance elements related to a given task [8], [10]. In the second case, indoor object segmentation based on depth data usually rely on finding planar surfaces and objects lying on it. Those methods can accurately detect objects on tables or floor, but are limited by the sensor quality, strong geometrical considerations (size or distance to the objects) and require more computation time than bottom-up saliency maps.

[1]U2IS, ENSTA ParisTech, Inria FLOWERS team, Université Paris-Saclay, 828 bd des Maréchaux, 91762 Palaiseau cedex France celine.craye@ensta-paritech.fr

[2]Thales - SIX - Theresis - VisionLab 1, avenue Augustin Fresnel, 91767 Palaiseau, France celine.craye@thalesgroup.com

So far, these visual attention approaches are mostly used as black boxes and are not learned (although sometimes refined) directly during the exploration. Machine learning approaches such as deep-learning are also trained offline, often in a fully supervised setup, and thus may not be adapted to different or dynamic environments. A more adaptive approach would be to learn and adapt to an environment directly on a robot, in an incremental and autonomous way. Learning would then specialize for a specific environment, but would be constantly improved and remain flexible to any change or novelty. Such learning would need an exploration strategy to gather relevant training samples.

Active exploration by robots can be done in many different ways depending on the task and available hardware. On mobile robots, predefined path plans [17], navigation graphs [13], or frontier-based explorations [12] can drive the robot's exploration. When equipped with a camera with zooming capabilities, [3], [13], [18], the perception of the robot can be further improved by moving the camera to relevant portions of the visual field. In the context of task learning, Oudeyer *et al.* proposed an exploration strategy based on intrinsic motivation such as learning progress [19] and competence progress [2]. In this approach, the robot is not trying to cover the whole environment or to improve its perception for a specific task, but is focusing on areas where learning is the most efficient. This way, the environment is not uniformly covered, but is learned more efficiently based on the current state of knowledge rather than on extrinsic criteria.

In this paper, we propose two contributions. First, we present an algorithm that can learn object-based visual saliency in an incremental and autonomous manner, directly within a robot's environment. Second, we suggest an exploration strategy driven by learning progress to allow the robot to autonomously learn about its environment faster and better. For that, we use an object segmentation algorithm from depth data that provides reliable and accurate, but partial and computationally expensive estimation of the saliency. From its result, we train a classifier that learns the visual RGB aspect of the discovered salient elements. Thus, the classifier is able to estimate the saliency of each pixel based on the RGB component only and reconstruct a saliency map faster and in less restrictive conditions than object segmentation does. It is for example capable of predicting object existence in areas where depth information is not available. We then use an intrinsically motivated exploration strategy based on learning progress to speed up and improve the learning quality. This paper extends previous work [6] in terms of segmentation algorithm and exploration strategy. In [6], the segmentation was obtained from depth data extracted only in

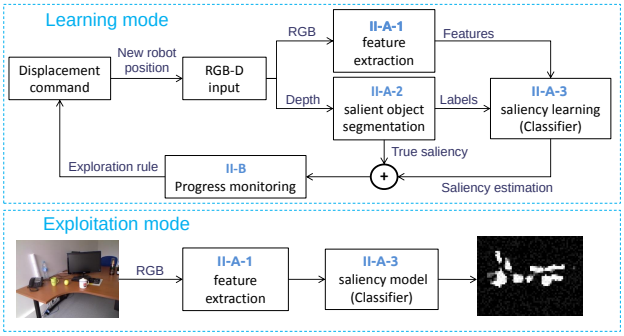


Fig. 1. General architecture of our system along with reference section for each block

a small portion of the input image (the fovea). We here use another segmentation technique that is applied on the entire image. Moreover, in previous work, the exploration strategy was used to select the position of the fovea in each input image, without considering the displacement of the robot in its environment. We now go one step further by selecting the next displacement of a mobile robot in its environment instead of the focus point in the current image.

II. Proposed approach

Figure 1 presents the general architecture of our system along with the corresponding section for each block. In a learning stage, the system learns the visual (RGB) aspect of salient elements within their context using a depth-based object detector as a supervision signal. We use an adapted version of the *Intelligent Adaptive Curiosity* (IAC) [19] to drive the exploration of the environment and make saliency learning faster and more efficient. Once exploration and learning is finished, we exploit the model to generate environment specific saliency maps using only the RGB image. The next two sections describe the overall method by describing the incremental saliency learning approach and the exploration strategy based on IAC.

A. Saliency learning

We define salient elements as being objects of the environment that are lying on planar surfaces (typically tables or floor), with a range size between 10 and 150 centimeters. The saliency learning is made possible by the interaction of three modules described below.

1) A feature extraction applied to the RGB image that encodes the color of each pixel and its neighborhood at different scales, averaged using superpixels. The method is more extensively explained in [6]. The feature extractor is applied on the whole input frame, and returns a 39 dimensions feature vector for each pixel. The feature extraction is much faster to compute than the geometric segmentation, and is available everywhere.
2) A segmentation algorithm that detects objects lying on planar surfaces is applied on the depth map. We use an adapted version of the method proposed by [5]. The depth map is turned into a point cloud for geometrical calculation. First, the main plane of the input point cloud, supposed to be the floor plane is detected and labeled as *not salient*. Then, big planes that are perpendicular to the ground are filtered and also labeled as *not salient*, as they are likely to be walls. Remaining points are then grouped by a k-d tree clustering and represent object candidates. To avoid false positives as much as possible, we categorize as *unknown* clusters that are either too small, too larger, or having a contact with the border of the frame. Remaining clusters are considered as *salient* objects. The point cloud is then converted back to a segmentation mask of the image frame based on the obtained labels (see Fig. 2, second row for illustration).
3) A classifier that is continuously updated based on the saliency labels provided by the segmentation mask and the corresponding RGB features. Each pixel of the input image is associated with a feature vector from the feature extractor and a label from the segmentation. We train our classifier with the feature-label samples in order to predict the saliency of a given pixel. We only train the classifier with data that is classified as *salient* or *not salient*. Pixels classified as *unknown* are not considered. The classifier used in our implementation is a random forest, which is not designed for online training. No available version [15], [22] of online random forest was satisfying in terms of speed and performance, so we adapted the offline version to make re-training fast enough. To this end, a limited number of samples from each new frame is used to update the classifier: for each new frame, we randomly pick up 700 pixels and add the corresponding data to a dataset cumulated from the beginning of the sequence. Then, we update the classifier by only re-training a small fraction of the forest at a time: we randomly select 4 trees among the 30 in the forest, and we retrain those tree with 70% of the dataset cumulated from the beginning of the sequence. Lastly we restrict the size of the cumulated dataset to 100000 samples. If the dataset exceeds this size, we randomly remove samples to meet the maximum size requirement. As a result, after each update, the classifier is able to estimate the saliency of an input based on the model trained with the previous observations, and the RGB image only.

In the exploitation stage, the saliency map is constructed as follows: RGB features are extracted for each pixel of the input and sent to the classifier. For each pixel, the classifier outputs a score between 0 and 1 that estimates the saliency. The classifier score is then associated to each pixel of the input image to reconstruct the saliency map.

B. IAC-based exploration

Exploration to gather training samples has a critical impact on the learning quality and efficiency. In the scope of autonomous and lifelong learning, an exploration strategy is necessary to guide learning and focus on areas where

learning is neither trivial nor impossible. Making sure that learning is still possible avoids decreasing in the learning quality because of irrelevant samples, and speeds up the overall learning rate by focusing on appropriate tasks first.

Our exploration strategy is based on an adapted version of the IAC (Intelligent Adaptive Curiosity) algorithm [19]. IAC is a method to drive a robot's actions toward situations that maximize the learning progress. This algorithm makes the agent focus on cases that are neither too easy nor too hard based on the learning state, so that progress are constantly made and no time is wasted in unlearnable situations. Moreover, progress maximization has been found to be an optimal strategy to learn in a limited amount of time [16]. IAC has been manly used to learn a mapping between motor commands and sensory feedbacks. To our knowledge, the algorithm has not been applied so far for pure vision problem, or used for robotics navigation.

IAC's key components are:

- a learner that learns to make predictions about the environment. In our case, the learner is the random forest module described in section II
- a separation of the exploration space into regions. In our case, the regions are divided based on positions and orientations of the robot.
- a meta-learner monitoring the learning rate in each region, and estimating the progress in each of them.

IAC suggests that the next action to be taken by the robot should be randomly picked up in the region having the highest learning progress. We use a similar approach to drive the robot in its environment: the environment (typically the room to explore) is arbitrarily divided into regions. After evaluating the progress in each of them, a position and orientation of the robot is randomly selected in the most progressing region. The robot next moves to this new position, obtains a new RGB-D input there, updates its saliency model and updates the progress estimation.

In our approach, the meta-learner stores for each region a history of the error rate based on the differences between estimated saliency (from the learner estimation) and observed saliency (from the segmentation algorithm). When a new frame is acquired in region i after t observations in the region, the meta-learner of region i is updated by adding the current error rate $Err_i(t)$ to the history:

$$Err_i(t) = 1 - F_1(C(O,E)) = 1 - \frac{2tp}{2tp + fp + fn} \quad (1)$$

where $F_1(C(O,E))$ is the F_1 score of the confusion matrix $C(O,E)$ based on the observed O and estimated E saliency maps, tp, fp and fn are the true positives, false positives and false negatives, from the confusion matrix. We use the F_1 score as our error metrics for Err_i, because *not salient* pixels are representing more than 80% of the samples, making accuracy inappropriate for error estimation.

An estimation of the learning progress in region i, is obtained by a linear regression of the error rate history Err_i over the last θ samples:

$$\begin{pmatrix} Err_i(t-\theta) \\ \vdots \\ Err_i(t) \end{pmatrix} = \beta_i(t) \times \begin{pmatrix} t-\theta \\ \vdots \\ t \end{pmatrix} + \begin{pmatrix} \epsilon(t-\theta) \\ \vdots \\ \epsilon(t) \end{pmatrix} \quad (2)$$

with $\epsilon(t)$ the residual error. The learning progress LP_i in region i is defined as the derivative of the learning curve (or the opposite of the error rate) in i. Therefore, we have

$$LP_i(t) = -\beta_i(t) \quad (3)$$

In the original version of IAC, the region to be explored next is the one with highest learning progress. This strategy is optimal only if taking a specific action has a negligible cost. In the case where actions are displacements of a robot from one location to another, it is clear that exploring in the same area for a few iteration may be more efficient than constantly making long displacement to the most progressing areas. To make learning progress and action selection better suited to our case, we force exploration to stay in the same region until N frames are obtained. The exploration procedure is then as follows: In a given region, we randomly select N positions (we found 10 to be a good trade-off in our experiments), and get an associated frames for each of them. We update at each new frame the learner and meta-learner. After N frames, we select the next region i to be explored with a probability that is proportional to LP_i 75% of the time, and randomly 25% of the time. We then move to this region and get N new frames from there.

III. Experimental results

To validate the efficiency of our approach, we used two different datasets.

The first one was collected from a Pioneer 3DX robot, with a Kinect RGB-D camera mounted at 1 meter from the ground and tilted slightly downward. We manually controlled the robot in a room containing several objects such as chairs, desks, or trash bins. We recorded a 5 minutes length video sequence in which a large number of views of the room were captured. Then, we moved the objects in the room and recorded another video so as to get a sequence for learning and one for testing. On the testing sequence, we manually labeled the salient elements for about 100 frames in order to evaluate the prediction capability of our system.

The second one is a publicly available dataset called *RGB-D scenes dataset* [14], composed with 8 video sequences of indoor scenes of everyday-life objects on tables. Objects are labeled in bounding boxes, which is not accurate enough for our evaluation. Therefore, we use the segmentation result of 100 frames and clean them up to obtain a ground truth. Those frames are removed from the training dataset.

The experiments were run on Ubuntu 10.12 with an Intel Core i3-3240, CPU at 3.4GHz quadcore processor. To learn saliency, several configurations were tested and compared. Frames of the sequence were either presented in a chronological order, random order, or based on learning progress. Section III-A provides details about the efficiency of the

resulting saliency itself, whereas section III-B describes the different strategies and their efficiency to speed up learning.

A. Saliency learning evaluation

Before presenting the incremental learning progression, we first analyze the final performance reached when enough samples are used to train the classifier. In our case, we found that about 1000 frames were enough to obtain the best possible accuracy on both datasets. However, for clarity, the evaluation in this section is done with a saliency model that was learned from the entire training sequence and evaluated on the testing sequence. We first demonstrate the benefit of using a learned saliency approach compared to the original segmentation only. Figure 2 illustrates a few frames from the sequence together with the saliency and the segmentation results.

From these samples, we observe that the segmentation algorithm is such that nothing salient can be detected further than 4 meters away. On the other hand, saliency estimation is applied on the whole image and the generalization capability of the classifier makes it possible to detect salient objects at more than four meters. Second, reflective surfaces are often hard to detect by the Kinect sensor. However, the aspect of salient reflective objects can be partially learned and fully retrieved based on the RGB data only. Third, the segmentation algorithm is very restrictive and is often not able to detect salient elements if they are in contact with a border of the image or badly captured by the Kinect. On the contrary, the saliency algorithm provides an estimate of saliency even if the object is partially cut, occluded, or captured with a poor image quality. Last, the segmentation processes input at 0.5Hz to 1.5Hz, depending on the geometrical complexity of the input. The saliency map estimation is processed at 8Hz, which makes it much more computationally efficient. Once exploration is finished, the segmentation can be disabled and saliency can be estimated at 8Hz for further processing (object localization for example).

To demonstrate the accuracy of our saliency estimation, we select three related saliency algorithms and compute the ROC curves for each method on both datasets. BMS [23] and GBVS [11] are among the most accurate RGB saliency methods according to the MIT *saliency benchmark* [4]. BMS is used with parameters that highlight salient objects rather than salient fixations. In addition, we use the new version of the VOCUS2 algorithm [9] along with the configuration file dedicated to the task of object detection in cluttered scenes (top-down saliency). On the robot sequence, we also evaluate the segmentation performance. Regions that are not labeled as *salient* or *not salient* are replaced by random noise between 0 and 1. Results are displayed in figure 3.

Based on the ROC evaluation, our method significantly outperforms the evaluated bottom-up and top-down techniques on both datasets, which was expected because it is trained specifically for the environment. We can also observe that most of the evaluated techniques outperforms the depth segmentation. This is because segmentation only returns a saliency result when this information is available. As a result, a large portion of the input frames is neither salient nor not salient and therefore estimated as noise in the ROC evaluation. For the same reason, when available, the segmentation hardly ever makes mistake about saliency prediction, which explains why the true positive rate is much higher than other methods when the false positive rate is very low.

In figure 4, a comparison between the methods is presented. First, our method provides an estimate of the shape and size of the salient object, which is not the case of techniques such as GBVS that are more stimuli-based. Second, the segmentation, that is based on geometrical consideration, avoids the detection of distractors that are visually salient (windows, trees outside, red power outlet) but irrelevant for our tasks. Last, it enhances elements that are not naturally salient (mobile container, desk) but consistent with our definition.

B. Exploration strategy

We now look at the evolution of the saliency quality during incremental learning. Figure 5 first shows a qualitative example of the evolution of the saliency at a given point of view, while the sequence is used in chronological order for training the classifier. We can observe the generalization capability because even before the seat was observed (in frame 400), the classifier is already able to recognize it as a salient element, because it has already learned a partial model of the background.

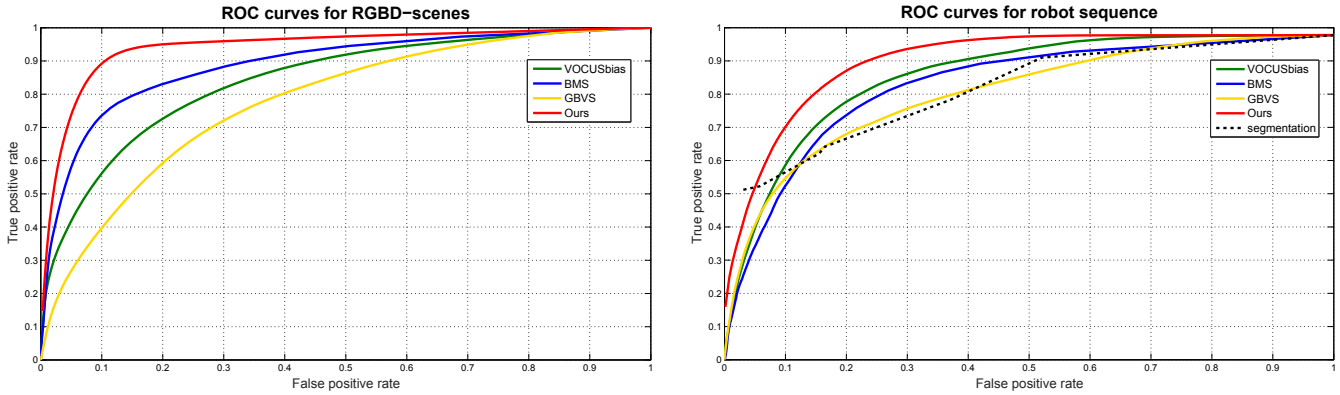
To evaluate the exploration strategy proposed in Section II-B, we use the two datasets by simulating displacements of the robot. Instead of selecting a random position/orientation in a region and physically moving to this location, we associate each frame of the dataset with a position/orientation (and, as a result, a region), and we randomly pick frames in the regions selected by IAC. For the two sequences recorded in our laboratory, the positions and orientations of the robot were obtained by a SLAM algorithm. We then arbitrarily defined 16 regions based on the position and orientation of the robot in the room: the positions were segmented in 4 equal regions based on the center of the room map, and the orientations were segmented in 4 equal cluster inside each region (see Figure 8). For *RGB-D scenes*, we used the sequence *table small 2* only. The video sequences are provided without any localization information. However, the trajectory of the acquisition sensor is such that each point of view is seen only once in the sequence. We then created 5 regions by dividing the video into five sub-sequences of equal length.

To demonstrate the benefits of exploring the environment using IAC, we compare the performance of the system when positions of the robot are selected with IAC, randomly, and following the chronological order of the sequence. In practice, selecting a position of the robot is equivalent in our case with selecting a frame from the dataset.

The performance of the system was evaluated using the evolution of the *overall error rate* of the system. Based on the reference frames on which a ground truth is available, we



Fig. 2. Saliency and segmentation results for a few examples. First row: the RGB image. Second row: the segmentation result, obtained from the depth input only. Grey areas of the segmentation images are pixels labeled as *unavailable* (dark gray), or *undetermined* (light gray). Third row: the saliency result, obtained from RGB features only. The classifier outputs for each pixel a fuzzy score between 0 (*not salient*) and 1 (*salient*). Note that the classifier is able to generalize so as to retrieve the saliency of data that is unavailable in the depth input.



(a) (b)

Fig. 3. ROC curves of several approaches on the *RGB-D scenes* dataset (a) and on the robot sequence (b).

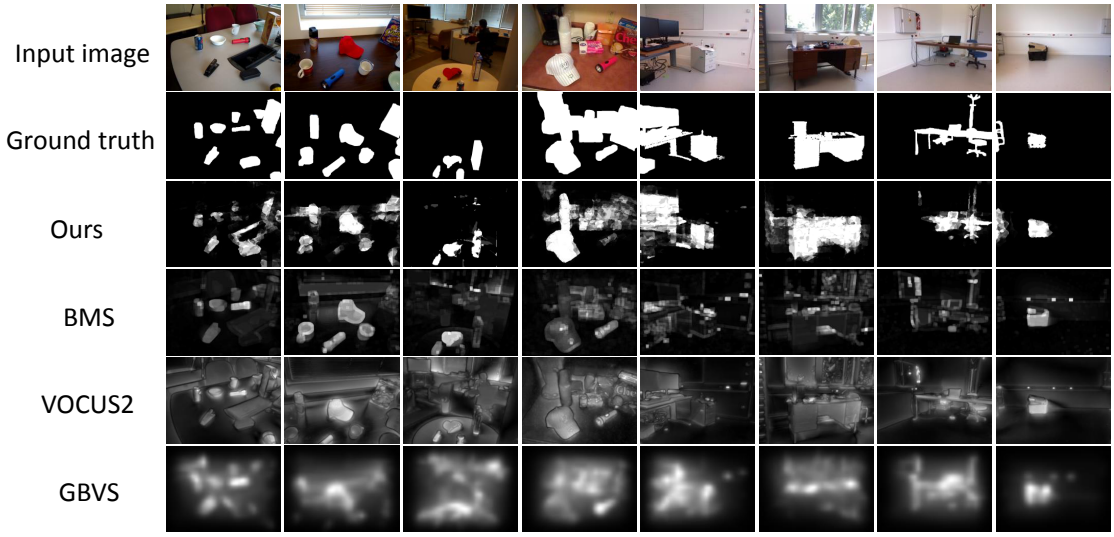


Fig. 4. Sample saliency maps for the five evaluated methods.

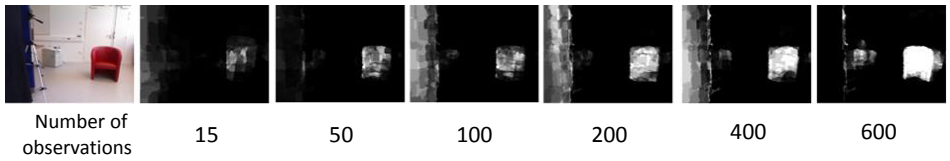
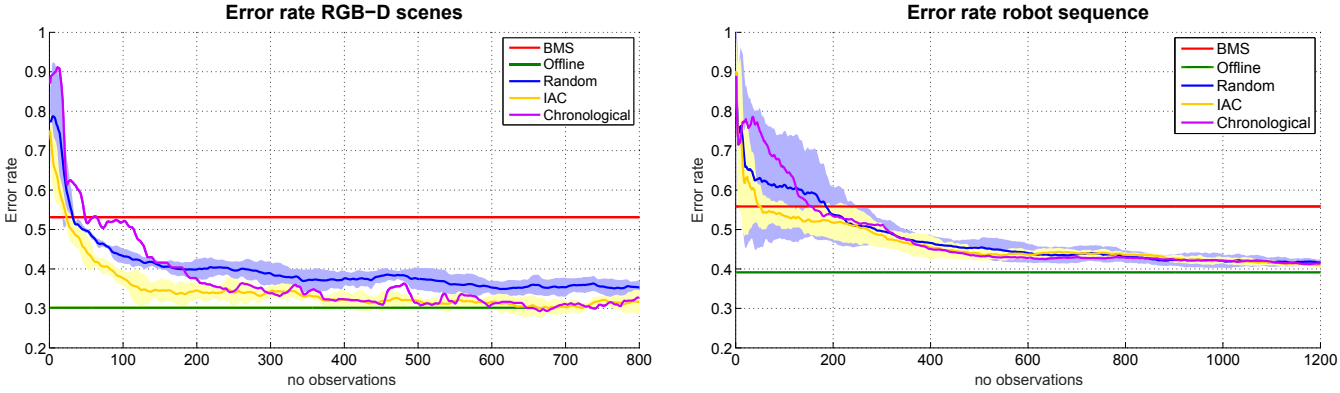
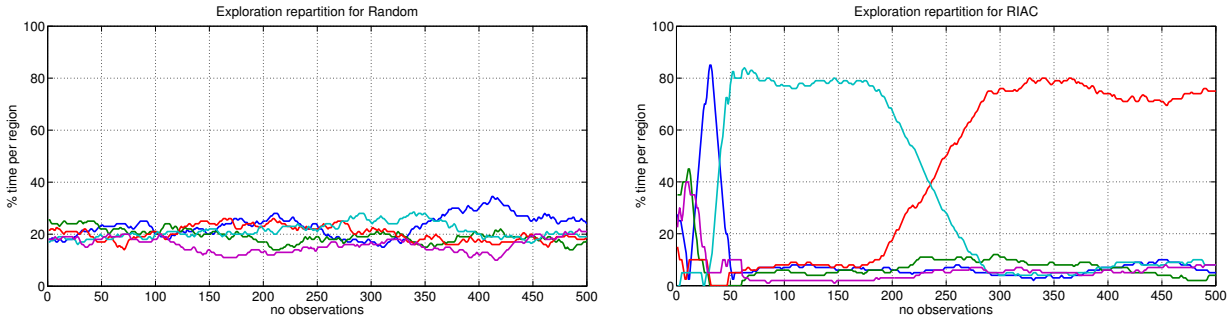


Fig. 5. Evolution of the saliency as number of observation increases. In this example, the frame were learned in chronological order, and the seat was observed for the first time only after 400 frames.



(a) (b)

Fig. 6. Overall error rate and variance evolution as new observations are obtained. Note that BMS (in red) is a static model of saliency so that it does not evolve in time. Results are obtained both on (a) *table small 2* sequence and (b) on the robot sequence



(a) (b)

Fig. 7. Time spend for (a) random frame selection and (b) IAC frame selection

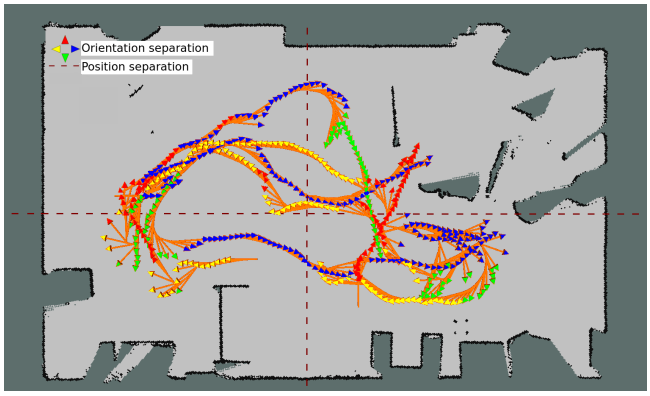


Fig. 8. Trajectory of the robot in the lab for one of the two sequences and division of the space into regions. Arrows represent a position/orientation at which an RGB-D frame is available. Dashed lines show the separation of the space based on position. Color of the arrow show the division based on orientation. In total, 16 regions are available.

compare the estimated saliency map for all of these frames with the available ground truth. Using the formula provided by equation 1 for each frame and taking the average error, we obtain the *overall error rate*. Note that the *overall error rate* differs from the *region error rate* used to determine the learning progress in Section II-B. The *overall error rate* is an extrinsic metrics used to evaluate the performance of the system, whereas the *region error rate* is intrinsic (based on segmentation rather than ground truth) and is used to get an estimate of the error in each region. At each new observation, we evaluate the current *overall error rate* and compare its evolution in time for each exploration strategy.

For IAC and random frame selection, we run the experiment 10 times and consider the average and variance over those sequences. As a reference, we also display the *overall error rate* obtained with a model trained offline with all samples of the entire sequences, as well as the performance

of the BMS [23] saliency maps. Results are displayed in Figure 6. As expected, the offline model is a lower bound of the error rate, and the system tends to reach this limit after a certain number of observations, whatever the exploration strategy. Our method rapidly outperforms BMS when enough observations are obtained. The chronological exploration is slower to converge than random exploration at the beginning. Last, IAC seems to be the exploration strategy that provides the faster learning rate. Note that the difference between random and IAC-based exploration is less significant on the robot sequence, but the error variance of IAC is much lower than the one for random exploration.

To get a better insight of the selection of the region and the proportion of time spend in each region, we represent in Figure 7 the proportion of time spent in each region at each new observation. The curves were obtained from the *table small 2* sequence, each curve representing the percentage of time spent in a given region. As opposed to random selection, where the time spend in each region is almost constant, IAC has a clear evolution of the most visited region. Starting with regions where error rate is decreasing fast (*i.e.* where progress is high), it then spends much more time on regions where progress is slower.

IV. Conclusion and future work

In this paper, we have presented an approach to incrementally learn visual saliency, using an exploration strategy based on learning progress. Using a reliable, yet restrictive and slow segmentation of salient elements, we construct a model of saliency that is much faster to compute, and getting better as new observations are obtained. This method shows good performance in the case of detecting objects lying on planar surfaces, as a clear geometrical definition can describe these situations. In this case, our results outperform state-of-the-art saliency approaches. To allow the robot to autonomously discover and learn about its environment, we use an adapted version of IAC: we divide the exploration space into regions, and drive our exploration to spend more time on regions where progress is the highest. This type of exploration strategy makes learning faster and better than it would be with random exploration and enables lifelong learning.

In a future work, we would like to investigate further the exploration strategy based on learning progress. Indeed, the time spent by the robot for displacement from one region to another one is not taken into account and an optimal trade-off between displacement time and learning time should be found. We could also apply this framework with other definitions of saliency. Instead of a generic object segmentation, we could for example use objects detectors and specialize our saliency to find those objects within their environments.

References

[1] Haider Ali, Faisal Shafait, Eirini Giannakidou, Athena Vakali, Nadia Figueroa, Theodoros Varvadoukas, and Nikolaos Mavridis. Contextual object category recognition for rgb-d scene labeling. *Robotics and Autonomous Systems*, 62(2):241–256, 2014.

[2] Adrien Baranès and P-Y Oudeyer. R-iac: Robust intrinsically motivated exploration and active learning. *Autonomous Mental Development, IEEE Transactions on*, 1(3):155–169, 2009.

[3] Mårten Björkman and Danica Kragic. Active 3d scene segmentation and detection of unknown objects. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3114–3120. IEEE, 2010.

[4] Zoya Bylinskii, Tilke Judd, Frédo Durand, Aude Oliva, and Antonio Torralba. Mit saliency benchmark. http://saliency.mit.edu/.

[5] Louis-Charles Caron, David Filliat, and Alexander Gepperth. Neural network fusion of color, depth and location for object instance recognition on a mobile robot. In *Computer Vision-ECCV 2014 Workshops*, pages 791–805. Springer, 2014.

[6] Céline Craye, David Filliat, and Jean-François Goudou. Exploration strategies for incremental learning of object-based visual saliency. In *ICDL-EPIROB*, 2015.

[7] Erkut Erdem and Aykut Erdem. Visual saliency estimation by nonlinearly integrating features using region covariances. *Journal of vision*, 13(4):11, 2013.

[8] Simone Frintrop. *VOCUS: A visual attention system for object detection and goal-directed search*, volume 3899. Springer, 2006.

[9] Simone Frintrop, Thomas Werner, and Germán Martín García. Traditional saliency reloaded: A good old model in new shape. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 82–90, 2015.

[10] Fred H Hamker. The emergence of attention by population-based inference and its role in distributed processing and cognitive control of vision. *Computer Vision and Image Understanding*, 100(1):64–106, 2005.

[11] Jonathan Harel, Christof Koch, and Pietro Perona. Graph-based visual saliency. In *Advances in neural information processing systems*, pages 545–552, 2006.

[12] Islem Jebari, Stéphane Bazeille, and David Filliat. Combined vision and frontier-based exploration strategies for semantic mapping. In *Informatics in Control, Automation and Robotics*, pages 237–244. Springer, 2012.

[13] Danica Kragic. Object search and localization for an indoor mobile robot. *CIT. Journal of Computing and Information Technology*, 17(1):67–80, 2009.

[14] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824. IEEE, 2011.

[15] Balaji Lakshminarayanan, Daniel M Roy, and Yee Whye Teh. Mondrian forests: Efficient online random forests. In *Advances in Neural Information Processing Systems*, pages 3140–3148, 2014.

[16] Manuel Lopes and P-Y Oudeyer. The strategic student approach for life-long exploration and learning. In *Development and Learning and Epigenetic Robotics (ICDL), 2012 IEEE International Conference on*, pages 1–8. IEEE, 2012.

[17] David Meger, Per-Erik Forssén, Kevin Lai, Scott Helmer, Sancho McCann, Tristram Southey, Matthew Baumann, James J Little, and David G Lowe. Curious george: An attentive semantic robot. *Robotics and Autonomous Systems*, 56(6):503–511, 2008.

[18] Silviu Minut and Sridhar Mahadevan. A reinforcement learning model of selective visual attention. In *Proceedings of the fifth international conference on Autonomous agents*, pages 457–464. ACM, 2001.

[19] P-Y Oudeyer, Frédéric Kaplan, and Verena Vanessa Hafner. Intrinsic motivation systems for autonomous mental development. *Evolutionary Computation, IEEE Transactions on*, 11(2):265–286, 2007.

[20] Houwen Peng, Bing Li, Weihua Xiong, Weiming Hu, and Rongrong Ji. Rgbd salient object detection: A benchmark and algorithms. In *Computer Vision–ECCV 2014*, pages 92–109. Springer, 2014.

[21] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*, 2014.

[22] Amir Saffari, Christian Leistner, Jakob Santner, Martin Godec, and Horst Bischof. On-line random forests. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1393–1400. IEEE, 2009.

[23] Jianming Zhang and Stan Sclaroff. Saliency detection: a boolean map approach. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 153–160. IEEE, 2013.