Unsupervised and online non-stationary obstacle discovery and modeling using a laser range finder

Guillaume Duceux, David Filliat

Unsupervised and online non-stationary obstacle discovery and modeling using a laser range finder

G. Duceux, D. Filliat

***Abstract*— Using laser range finders has shown its efficiency to perform mapping and navigation for mobile robots. However, most of existing methods assume a mostly static world and filter away dynamic aspects while those dynamic aspects are often caused by non-stationary objects which may be important for the robot task. We propose an approach that makes it possible to detect, learn and recognize these objects through a multi-view model, using only a planar laser range finder. We show using a supervised approach that despite the limited information provided by the sensor, it is possible to recognize efficiently up to 22 different object, with a low computing cost while taking advantage of the large field of view of the sensor. We also propose an online, incremental and unsupervised approach that make it possible to continuously discover and learn all kind of dynamic elements encountered by the robot including people and objects.**

I. Introduction

Simultaneous Localization And Mapping (SLAM) techniques with laser range finders have proven to be efficient for indoor navigation [18]. However, those techniques usually assume a static environment, relying on a world model that do not provide semantic knowledge about obstacles and ignore or filter non-stationary objects. These objects are often things that are interesting for the robot tasks, such as doors, chairs or people moving. In order to perform more complex navigation such as opening a door or moving a chair out of the way, the robot should therefore be able to recognize them. This problem is closely related to semantic mapping [15] where the purpose is to build a map of the environment with higher semantic knowledge such as rooms and objects.

It is possible to give the robot a prior knowledge of certain objects using supervised learning techniques, but it is impractical to do so for all possible dynamic objects it will encounter if we imagine a long-term use of robots in homes. Therefore it is interesting to give the robot the ability to learn and model these objects on-line while it is performing other tasks. Since those objects are dynamic, it is possible to use a change detection system to discover them, and then use the learned knowledge to recognize them later. This problem relates to object discovery approaches [20] which involve detecting changes between some inputs (images, maps, range data), computing a description of this changing part, and creating an object model by some clustering method. Typical drawbacks of object discovery methods are to be off-line and to be unable to handle all the objects, especially those with changing shapes (like people).

G. Duceux (duceux@ensta.fr) and D. Filliat are with the ENSTA ParisTech - INRIA FLOWERS Team, Computer Science and System Engineering Laboratory, ENSTA ParisTech, 828 boulevard des Marechaux, 91762 Palaiseau, France.

Most of the work on object discovery and semantic mapping has been done using sensors such as 3D lasers [15], color cameras [20] or RGB-D cameras [7] or working with 3D maps [9]. Although those sensors provide a rich information, the computation involved is often heavy. Less work has been done using 2D laser range finder [14]. While being aware that this sensor limits the type of objects that can be recognized and the potential performance of the system, we argue that a number of useful objects can be recognized, and that even in case of confusion, it can provide a good prior for another more computationally complex object recognition based on a richer sensor such as an RGB-D camera.

In this paper we therefore present an object discovery method based on laser data in a navigation context. We assume that the robot starts by exploring an environment and build an occupancy grid map using SLAM techniques. This map will contain most of the static elements of the environment, along with some dynamic elements, such as chairs, that remain still during mapping. Afterwards, as the robot navigates in the environment, we use the map for localization, detect dynamic objects that are inconsistent with the map, and build multi-view models in order to categorize and recognize them. The multi-view model enables to deal with different points of view as well as changing shapes, and therefore adapts to the various kind of dynamic objects such as objects, doors, animals or humans. Furthermore, the map is updated in order to filter out dynamic aspects and to include initially unknown places revealed by the moving objects. As such the robot builds a represention of the world consisting of both a map and object models to represent respectively the static and dynamic aspects.

The paper is structured as follows. First, in section II we give an overview of related works. Then, we describe our approach in section III and present the experimental results in section IV before discussing the relevance of our method in section V.

II. Related Work

Object discovery, following the definition of e.g. [9], usually include two steps, a first step for detecting candidate objects using different approaches such as a generic object segmentation, detecting novelty or re-occurring patterns, and a second step performing unsupervised learning to model the objects.

A. Object detection

Detection can be made using differences between maps taken at different times. The work in [9] is based on 3D

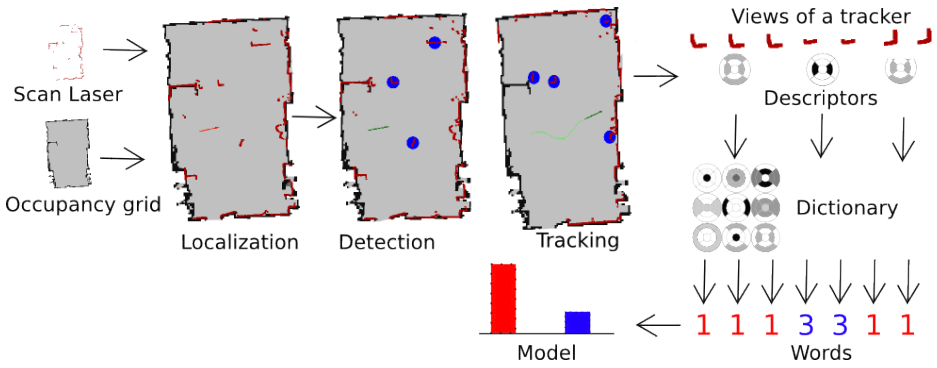

Fig. 1. Object modelling process proposed in this paper, from robot localization to object model.

maps represented as point clouds. They compare two maps of the same environment and use the map differences as possible objects. A similar approach is used by [3] using 2D occupancy grid maps.

Others work directly with sensor data. For example, [13], [7] and [1] use plane segmentation based detection on depth images to discover or search objects. The object are therefore supposed to stand out of flat surfaces such as the floor or a table. In computer vision, as the task of discovering object without prior knowledge is difficult, statistical methods are often used over large datasets to find re-occurring patterns (see [20] for a review). Concerning 2D range data, [14] uses a filtering method on range data localized by SLAM techniques to discover novelty. This is the approach we used, as it is simple and efficient enough for our purposes.

B. Object modelling and recognition

There are numerous approaches to model and recognize objects. Using range data, there are mainly three:

The first is to use registration or scan matching to generate and recognize geometrical models. In [9] and [14], they align surfaces of an object in 3D and 2D range data respectively. By aligning those surfaces, they obtain a model of the object consisting of a point cloud as it would be seen by the sensor if it could see the entire object. Those approaches are very susceptible to noise in data sensor, and are not well suited for modelling object with changing shapes like people.

The second is to extract invariant local features from observations and to differentiate objects based on their set of features. The work of [6] uses local shape descriptors with Latent Dirichlet Distribution on 3D range data. Their method is unsupervised but not on-line and assumes knowledge of the number of object in a scene is known. An object is represented in this case as a distribution of local surface shapes. This kind of object model is more robust to noise in sensor data and change in object appearance. They are usually faster and more efficient than geometrical models to learn and to recognize. The use of local features has also been widely studied by the computer vision community. There is a wide variety of those methods [4] which require the encoding of certain properties of the appearance into descriptors, and the clustering of those properties.

The third is to create multi-view object models regrouping the views of an object from different viewpoints or different times. This approach is used in [12] based on vision, taking advantage of object manipulation by a robot to gather different views of the objects. Using tracking techniques to put together different views have also been used in [14] using laser scans. In [16], the authors model views and their associated metadata (segments, positions, time, etc...) in a graph. Rich object models are obtained by clustering this graph. In order to memorize the views, an associated descriptor can be used. Many of these exist in vision or 3D, but far fewer for 2D range data. Nevertheless, [19] shows that it is possible to use descriptors with laser range finder, applied to place recognition in their case.

III. Proposed method

In this paper, an appearance of an object, which we call *view*, consists of a set of points belonging to one object obtained from a single laser sensor reading. As these views have different number of points depending on the distance of the object, a view is encoded into a shape descriptor of constant size that is invariant to distance and rotation. Descriptors are then clustered into a dictionary which associate for every descriptor a label called *word* following the terminology of the Bag of Words approach [17]. An object is therefore modelled as a set of possible words following this approach. Figure 1 and the accompanying video present our overall approach described in the following.

A. Simultaneous localization and mapping

Our system needs to build an occupancy grid map of the static part of the environment, to update it and to localize the robot inside during several separated runs. This grid, called the Static Map, is maintained from one run to another. For each run, the Hector SLAM algorithm [11] is used, starting from an empty map each time, to produce an occupancy grid we call the Current Map. The output trajectory of Hector SLAM is localized in the Static Map using a particle filter. This approach results is a much more accurate position estimation in the static map than by localizing directly the robot in the static map using particle filtering. The reason is that when a lot of objects not represented in the Static

Fig. 2. A Static Map after the first run of the robot (left), and after several runs (right).

Map are present, the robot is better localized in the Current Map that contain these objects, and the resulting trajectory is more precise than if the robot is localized directly in the static map.

The static map is then updated using the laser scans in order to remove obstacles that disappeared and to fill previously unperceived areas revealed because an object has moved. However, we should not add new obstacles in known areas as we aim at mapping only the static part of the environment. To do so, standard occupancy grip mapping techniques are used to update only the unknown or occupied cells as well as their nearby unoccupied cells.

Figure 2 illustrates the map updating process. On the left is an occupancy grid of a room seen for the first time by the robot. On the right is the same area after several runs of the robot. We can see that the doors have been completely removed from the map as well as certain objects in the middle and against the walls. The shape of some furnitures however are still visible because they haven't been moved. A corridor and a second room have also been discovered by the robot.

B. Novelty Detection and tracking

The localization provides a set of laser endpoints localized in the Static Map reference frame that are noted x_i. An endpoint can either correspond to something static (wall, static furniture), or to an object that can move (chair, human, door). Points belonging to known static objects (mainly walls) should have a small distance to occupied cells in the Static Map. We therefore compare the distance d_i of each endpoint to the closest occupied cell to a threshold in order to detect points belonging to a non-stationary object.

The detected points are then clustered together given that they are at a certain distance *radius* of a cluster center, which is updated every time a point is added. After processing all detected points, non-detected points are added to the clusters with the same criterion but without updating the center this time. We set *radius* to $0.5m$ because most of the dynamic objects being considered are not wider than $1m$.

Those clusters form the detected objects, they are then tracked through time using the approach described in [14]. A descriptor (described below) is computed for every detection in every laser scan and tracking is used to put together descriptors belonging to the same objects to form the bag of word model.

C. Descriptor computation

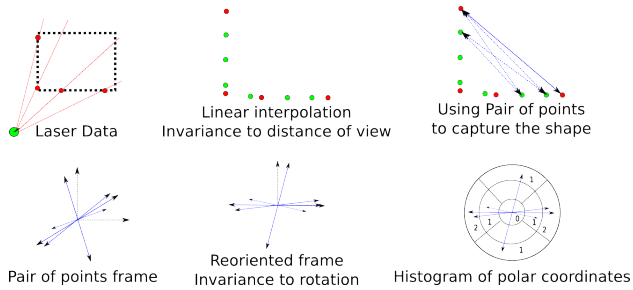

Fig. 3. Illustration of descriptor building steps.

As it is rather impractical to compare directly two sets of few points, a descriptor is computed for every view in order to compare them quickly and to achieve some invariances and a certain level of robustness regarding noise.

To be able to recognize non-stationary object, invariance from point of view is required. This means that if the robot sees the same part of an object, the resulting descriptor should be the same, independent of where the robot sees it. To achieve that, several steps are involved in the construction of the descriptor. Fig. 3 illustrates those steps. To construct the descriptor, we followed ideas from [2] and [10].

The detection provides a set of points representing a part of an object boundary. However, as the robot gets farther to the objects, fewer laser points will hit the object, and they would be more separated. The first step is therefore to re-sample the points with a fixed inter-distance in order to be invariant to the object distance. For each pair of successive points, we use a linear interpolation to generate new points at regular intervals (we use $1mm$), which leads to having almost the same amount of points when the object is seen from afar than up close.

For each pair of points in this set, the vector that goes from one point to the other is computed in polar coordinates $(r,\ \theta)$. The θ coordinate of those vectors is dependent on the rotation of the object in the map reference frame. In order to have invariance to rotation, a reference angle is computed as the maximum argument of the histogram of the θ coordinates distribution. For every vector, a new θ coordinate is computed relative to that angle of reference.

Finally, the descriptor is computed as an histogram of the polar coordinates of these vectors normalized by the number of points. The histogram is parametrized by the number of division of both the angular coordinate (in $[-\pi, \pi]$) and the distance coordinate (in $[0, 1]$m). Those parameters have an important role in the performances of the system.

To compare two descriptors, the Symmetric Chi-Square metric is used. A comparison of popular metrics [5] has

shown slightly better results in our case with this one. The distance is expressed as follows:

$$d_{\chi^2}(I,J) = \frac{1}{2}\sum_i \frac{(I_i - J_i)^2}{I_i + J_i} \quad (1)$$

with I and J two descriptors, I_i and J_i the i-th element of the descriptor I and J respectively.

D. Descriptor clustering

In order to have a compact representation of the objects, we follow the bag of words approach as described in the next section. For this, we need to compute a dictionary of descriptors obtained by clustering the perceived descriptors. We used the incremental method presented in [8]. In this method, a distance threshold is fixed to decide whether to create a new word in the dictionary or not, when a new descriptor is perceived. If the descriptor is far enough from all the words, it is used as the center of a new word, otherwise it is assigned to the closest word.

E. Object modeling

Objects are represented as bags of words, i.e., as histograms of occurrences of the different views from a tracker. An important problem is that the sampling of the views around the object will depend on the robot trajectory around it. As we want to construct the models online and be able to recognize objects with partial information, i.e., seen from only one side, we need to enforce a sampling of the views that will limit the dependency on the particular robot trajectory. To do so we filter descriptors during the construction of the model to increase chances of having similar models.

The filter comprises a condition on the relative position between the object and the robot and on the word being perceived. Indeed, since some objects might change shape, we can't filter only on the position. Therefore, we only add a word to the model if it is different from the previous one or if the relative position of the object has moved more than a given distance (we use $10cm$).

Two objects are compared using histogram intersection:

$$d_{\cap}(I,J) = \sum_i min(I_i, J_i) \quad (2)$$

with I, J two histogram being compared. Note that an object histogram is normalized by its number of elements. A comparison between popular similarities and distances metrics [5] has shown that although the difference is slight, the intersection gave the best results.

F. Incremental object recognition

When object are discovered incrementally during the robot navigation a mechanism is required to decide whether a newly perceived object is a novel object or a perception of an already known object. To do so, we keep in memory a set of object model clusters, each cluster corresponding to a single physical object.

Fig. 4. The 22 different objects in the database and their associated label for supervised tests. Two spiral trajectories have been recorded around each object.

When a new object is tracked, a model is built according to the previous section. When the tracking ends, the most similar model in the memory is found. If the corresponding similarity is higher than a threshold, the new model is added to the same cluster as the corresponding model. If not, a new cluster is created with the new model.

IV. Experimental results

We performed experiments using a Pioneer3 mobile robot equipped with an hokuyo utm-30lx laser range finder. The range finder has a precision of 0.03m from 0.1m to 10m and an angular resolution of 0.25 degrees.

In order to assess the quality of our object representation, we built a database consisting of 22 objects. To construct the database we moved the robot around the objects and recorded the trajectory and the laser data. Two trajectories were recorded by objects to ensure a separate training and test set. With this database, we performed experiments to set the different parameters using grid search and to evaluate the performances in an ideal case.

A. Descriptor evaluation

The first experiment was to control the efficiency of the descriptor regarding the invariance we were expecting. In order to do that, we generate a map of the words obtained as a function of the position of the robot.

Fig. 5 has been made with a dictionary threshold of 0.3. The descriptors has 6 bin on distances and 11 on angles. The figure shows that the expected invariances are achieved on objects with good response to laser sensor: objects 1, 7, 9, 18. Problems arise when far from an object as too few points are obtained from it, which limits the distance at which we can perceive it. Also, certain objects are not well perceived by laser range data, such as black colored objects and hard edges. In the later case, a smaller variation in the position of the robot produce different words, so a descriptor can still be computed and used in the recognition process but with less robustness. For this type of object, the model needs to contain

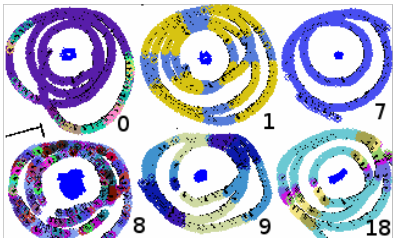

Fig. 5. Region with the same color represent where the robot has seen the same word in the dictionary. The number represents the object label. The regions formed are consistent with the invariances expected from the descriptor, except when laser data is too noisy for certain objects.

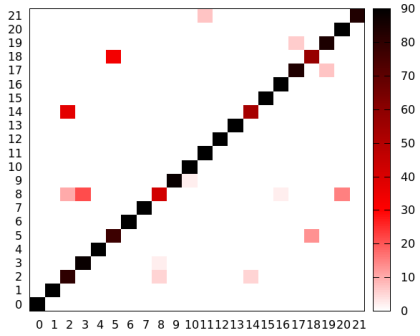

Fig. 6. Confusion matrix for object recognition with complete models.

words coming from several readings at the same place. That is why filtering of the repartition of views are made based on both relative position and the value of the word obtained. Finally, some objects are really noisy, such as 8 in figure 5 which is a moving human. But even in this case we will see that the corresponding bag of words is specific enough to recognize it.

B. Recognition with complete models

In a second experiment, we evaluated the performance of the recognition when seeing the objects completely, i.e., from all possible viewpoints. We constructed a set consisting of eight complete models of each object. In order to perform cross-validation, the set was randomly divided ten times into a training and a validation set. Each time, one model by object was randomly picked to go in the training set. The remaining models were put in the validation set. Each time a confusion matrix was computed. All the results were accumulated in a final confusion matrix shown by Fig. 6. A 89% global recognition rate was obtained.

Results show that the method works well with complete models. The false recognition are explained by the fact that, with a laser range finder, some objects are perceived as having very similar shape and size. For instance, the two chairs are more often confused as well as the box 18 with the box 5. However, most of the time the differences in size and shape are sufficient to avoid confusion. Lastly, the most confused object was the moving person. In fact, when moving, people's legs appearance for the laser sensor is highly variable, which cause high variation in the resulting models, hence more confusion.

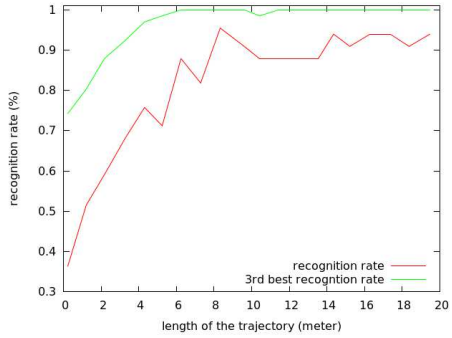

Fig. 7. Recognition rate as a function of the length of the trajectory.

C. Recognition with partial models

In a real application though, the robot should be able to recognize objects with partial models without performing a full circle around the object. In order to assess this in a controlled setup, we computed the recognition rate as a function of the length of the trajectory sampled from the same database. In this experiment, the training set still consists of one complete model for each object. The test set consists of randomly generated trajectories of varying length.

Fig. 7 shows the recognition results with two different criteria. For the first one (in red), we have considered an object as being recognized if the most similar object is the correct one. As expected, the more an object is perceived, the better it is recognized. Note that around 4 meters the recognition rate is already strong, which correspond to seeing about half of the object. This correspond to trajectories that the robot would have when avoiding an obstacle or passing by it. It suggests that recognition during the robot motion for another task could perform well.

In the second criteria (in green), we considered a recognition being successful if the right answer was in the three best score. The performances are clearly improved with a perfect recognition above 6 meters. This suggests that when the system is wrong on the identity of an object, it is not far off. For instance, when recognizing the black chair, we have seen that the system often confuses it with the blue one, but the similarity with the black chair would still be high. This result indicates that we could rely on this recognition as a good prior for mixing it with an algorithm using another modality.

D. Incremental learning

For this experiment, we tested incremental learning using trajectories sampled from the database in order to have a ground truth on the object identity and be able to assess the quality of the resulting clusters. In order to set the threshold for integration in a cluster, we studied the behavior of the clusters in memory when varying it. Fig. 8 shows that when the threshold is low, few clusters are created and they are mostly corrupted, i.e., they contain models from different real objects. On the other hand, when the threshold is too high, every tracking results into a cluster being added to the

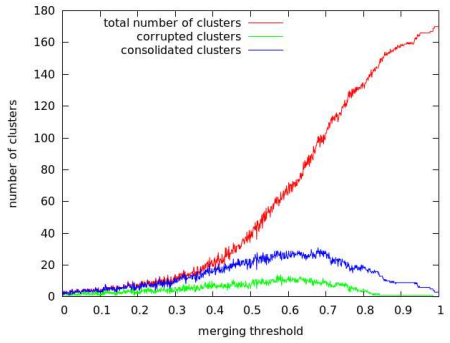

Fig. 8. Number of clusters as a function of the merging threshold. In red the number total of clusters in the memory. In blue, the number of clusters that have been updated successfully. In green, the number of corrupted clusters (cluster containing models coming from different objects).

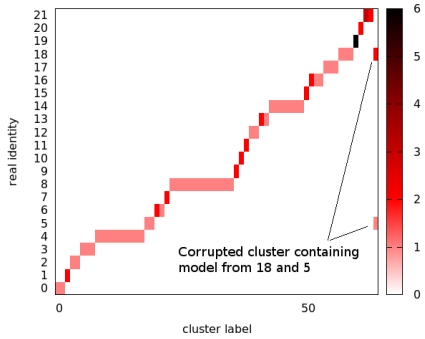

Fig. 9. Number of clusters in the memory by real objects identity and their size.

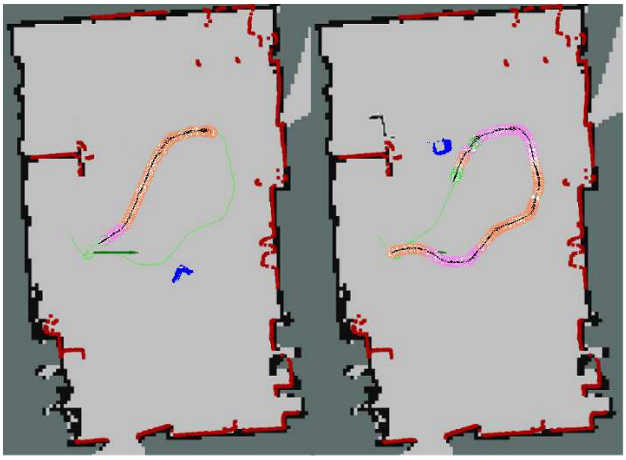

Fig. 10. Example of trajectory and models obtained. The green line represent the trajectory, the blues points are the laser reading on the considered object, the circles represent where the robot registered a word in the model for the object, the colors represent the words id.

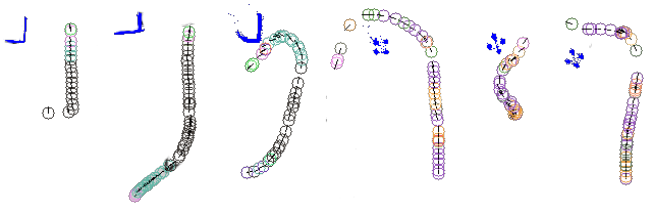

Fig. 11. Example of clusters obtained. On the left, three models coming from an armchair, on the right three models coming from a stool.

memory, and few cluster are updated. From these results, we choose a compromise and set the threshold to 0.75.

In order to see the resulting distribution of clusters in memory, we constructed figure 9. The database was split into 85 trackers with varying size (between 1 and 80 according to the live experiments, see section IV-E). The matrix was built by picking randomly one tracker from the dataset, and adding it to the memory as explained in section III-E, until the dataset was empty. For clarity reasons, the resulting set of clusters was ordered. In this case, we obtained 51 clusters with a single model and 13 clusters with multiple models, with 1 corrupted. Some objects resulted in few clusters in the memory (1, 7, 9, 10, 11, 15, 19, 20). Which means that the first time the object was seen, the resulting model was a good representation and that the object is easy to recognize. Other objects are more difficult to recognize from partial models and result in several clusters in the memory.

E. Live experiment

Finally the system was implemented on the robot in real-time as shown in the accompanying video. We used 8 different trajectories in a room containing 8 different objects that were moved between the robot trajectories. The system resulted in 125 different models, resulting in 16 clusters, among which 2 were corrupted.

On the trajectories that we studied we obtained a maximum of 81 words and an average of 16 words in each model, depending on the duration of the tracking of the objects. Fig. 10 shows an example of a trajectory with the associated words recorded with two different objects.

Figure 11 show two pure clusters of models constructed for two different objects. The words of each objects are plotted on the trajectory of the robot during its creation in order to show the diversity of the trajectories that make it possible to recognize an object.

For each implemented module, the mean computation time was recorded (table I). The code was written in C++ without particular optimization. Except the dictionary (which performs descriptor clustering) and the object manager module (which performs object modeling and recognition), the computation times are bounded. For the dictionary and the object manager, the computation times depend linearly on the number of words and clusters respectively.

Localizer	50 ms
Detector	0.7 ms
Tracking	0.1 ms
Descriptor	18 ms
Dictionary	1.2 ms with 440 words
Object manager	2.4 ms with 114 models
Total	72.4 ms after 10 runs

TABLE I

TABLE OF COMPUTATION TIMES.

V. Discussion

Obstacle discovery has been largely studied using vision sensors with various objects. In comparison, using a laser range finder limits it to objects that are on the ground. However, in a navigation task, most of the objects involved are perceived by this sensor. With this limitation, we have shown that it is possible to distinguish between a reasonable number of objects sufficient for common household setups and to perform unsupervised object discovery and recognition. Moreover, because the volume of data given by the sensor is small compared to vision, or 3D sensing, the resulting computations are less complex and can easily be performed in real-time.

Our modelling system, based on bag of words, creates multi-view models. The advantage of this approach is that it can handle objects changing shapes (such as humans or animals), as well as ill perceived objects such as dark or reflective surfaces whose appearance varies strongly even from close viewpoints. The use of descriptors instead of raw range data also improves robustness to noise. It avoids using scan matching which is difficult if there aren't many points to match when an object is perceived from far away.

Beside being able to recognize objects using supervised learning, the proposed approach can perform incremental and non-supervised object modelling, with reasonable performances even when the trajectories of the robot do not allow to perceive completely an object. This make it possible to adapt to an environment continuously and gather up information on new objects introduced in the environment, even while the robot is doing others tasks. Beyond our simple incremental approach, this information could be treated a posteriori with more complex techniques to refine the clustering of models, and to filter out noisy models.

Finally, our system could support and enhance a camera-based recognition system. The novelty detection and the tracking could help segmenting objects in an image. Moreover a laser range finder field of view is wider than those of a camera, so it is possible to recognize an object before the camera sees it and to orient the camera toward this object to help recognition. Even when the recognition result from the laser is uncertain, it could then be used as a prior to improve visual recognition.

VI. Conclusion and perspectives

We proposed an approach to perform dynamic object discovery, modelling and recognition using only a laser range finder commonly used to perform navigation tasks. We showed that in ideal conditions where the robot make complete circles around the objects, using a multi-view object model, it is possible to recognize up to 22 different objects of different types, including objects changing shapes such as humans walking. Applied to incremental object discovery, this same approach make it possible to create coherent object models without supervision.

As a laser range finder has a wider field of view, and less information to process, we believe that our approach is well suited to support and enhance a camera-based recognition system for a service robot. In future work, we therefore plan to use this system as a first stage to a second system following similar principles but based on a RGB-D camera to perform more efficient object recognition.

References

[1] A. Aydemir, M. Göbelbecker, A. Pronobis, K. Sjöö, and P. Jensfelt, *Plan-based Object Search and Exploration Using Semantic Spatial Knowledge in the Real World*, Proceedings of the 5th European Conference on Mobile Robots (ECMR'11), 2011.

[2] S. Belongie, J. Malik, and J. Puzicha, *Shape matching and object recognition using shape contexts*, IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002), no. 4, 509–522.

[3] R. Biswas, B. Limketkai, S. Sanner, and S. Thrun, *Towards object mapping in dynamic environments with mobile robots*, Proceedings of the Conference on Intelligent Robots and Systems (IROS) (Lausanne, Switzerland), 2002.

[4] Richard J. Campbell and Patrick J. Flynn, *A Survey Of Free-Form Object Representation and Recognition Techniques*, Computer Vision and Image Understanding **81** (2001), no. 2, 166–210.

[5] Sung-Hyuk Cha, *Comprehensive survey on distance/similarity measures between probability density functions*, International Journal of Mathematical Models and Methods in Applied Sciences **1** (2007), no. 4, 300–307.

[6] F. Endres, C. Plagemann, C. Stachniss, and W. Burgard, *Unsupervised discovery of object classes from range data using latent dirichlet allocation*, Proc. of Robotics: Science and Systems, June 2009.

[7] D Filliat, E Battesti, S Bazeille, G Duceux, A Gepperth, L Harrath, I Jebari, R Pereira, A Tapus, C Meyer, S Ieng, R Benosman, E Cizeron, J.-C. Mamanna, and B Pothier, *RGBD object recognition and visual texture classification for indoor semantic mapping*, Proceedings of the 4th International Conference on Technologies for Practical Robot Applications (TePRA), 2012.

[8] David Filliat, *A visual bag of words method for interactive qualitative localization and mapping*, Proceedings 2007 IEEE International Conference on Robotics and Automation (2007), 3921–3926.

[9] Evan Herbst, Peter Henry, Xiaofeng Ren, and Dieter Fox, *Toward object discovery and modeling via 3-d scene comparison.*, IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2011, pp. 2623–2629.

[10] Andrew Johnson, *Spin-images: A representation for 3-d surface matching*, Ph.D. thesis, Robotics Institute, Carnegie Mellon University, August 1997.

[11] Stefan Kohlbrecher, Oskar von Stryk, Johannes Meyer, and Uwe Klingauf, *A flexible and scalable SLAM system with full 3D motion estimation*, 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics, Ieee, 2011, pp. 155–160.

[12] Natalia Lyubova and David Filliat, *Developmental approach for interactive object discovery*, The 2012 International Joint Conference on Neural Networks (IJCNN) (2012), 1–7.

[13] Julian Mason and Bhaskara Marthi, *An object-based semantic world model for long-term change detection and semantic querying*, 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (2012), 3851–3858.

[14] Joseph Modayil and Benjamin Kuipers, *Bootstrap learning for object discovery*, Proceedings of the 2004 IEEE/RSJ Interniatonal Conference on Intelligent Robots and Systems **1** (2004), 742–747.

[15] Andreas Nüchter and Joachim Hertzberg, *Towards semantic maps for mobile robots*, Robot. Auton. Syst. **56** (2008), no. 11, 915–926.

[16] A Collet Romea, Bo Xiong, and Corina Gurau, *Exploiting Domain Knowledge for Object Discovery*, IEEE International Conference on Robotics and Automation (ICRA), 2013.

[17] J. Sivic and A. Zisserman, *Video google: a text retrieval approach to object matching in videos*, Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on, Oct 2003, pp. 1470–1477 vol.2.

[18] Sebastian Thrun, Wolfram Burgard, and Dieter Fox, *Probabilistic robotics (intelligent robotics and autonomous agents series)*, Intelligent robotics and autonomous agents, The MIT Press, 2005.

[19] G.D. Tipaldi and K.O. Arras, *Flirt - interest regions for 2d range data*, Robotics and Automation (ICRA), 2010 IEEE International Conference on (2010), 3616–3622.

[20] Tinne Tuytelaars, Christoph H. Lampert, Matthew B. Blaschko, and Wray Buntine, *Unsupervised Object Discovery: A Comparison*, International Journal of Computer Vision **88** (2009), no. 2, 284–302.