Multi-View Object Recognition Using Hidden Markov Model and Relative Positioning

Luigi F. Tedesco[1], Céline Craye[2], Jean-François Goudou[2] and David Filliat[1]

***Abstract*— Object recognition capability is an essential condition for giving autonomy to mobiles robots in human designed environment. However, achieving this goal by relying only on a visual representation of objects is a hard task. Fusing several sources of information to enhance representation can, therefore, be drastically helpful. In the context of object recognition for indoor environments, we present a procedure to integrate odometric data in the classification process. When fused with visual perception, odometry incorporates the notion of displacement between views that is able to overcome ambiguous points of view and associated misclassification. By observing objects from different perspectives and modeling the visual sequence as a Markovian process whose transitions are provided by odometry, our algorithm is able to copes with a sparse database, blurred images from motion and object spatial symmetry, to efficiently recognize objects and estimate the sequence of viewpoint at which they were observed. A multi-modal Kalman-based object tracking was also implemented in order to recognize multiple objects simultaneously. The approach was tested on a mobile robot and then compared with classic recognition. The significant gap between the two approaches is very promising and reinforces the proposal of blending odometry and recognition.**

I. INTRODUCTION

Object recognition for mobile robots in indoor cluttered environment is still an open problem. Recent progresses in computer vision techniques - typically deep learning based - have shown impressive results on complex image datasets [14]. However, those approach require heavy computation, large training datasets, and the recognition is based on RGB still images. On the other hand, mobile robots have the ability to move in their environment in order to modify and possibly improve their perception and representation. Relying on images without considering the displacement information is then a significant loss of information.

Objects and object-viewpoint recognition are among the most popular topics in robotics applications. Either used for environment exploration [4], [13], semantic navigation [8] or pose estimation for grasping [6], most of them typically rely on single-frame based recognition. Odometric information is also largely used on mobile robots in the context of object recognition, especially for the task of path planning [16], [19], tracking [24] or simultaneous localization and mapping (SLAM) [13]. Although odometry is widely used to localize and track the position of objects, it is rarely used in the process of recognizing them.

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

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

The use of RGB-D sensors in indoor environment has become very common because of their valuable information about the environment geometry. For this type of sensors, the object recognition pipeline is usually composed with a segmentation step followed by feature extraction before recognition itself. The segmentation step aims to differentiate objects from the background of raw images. By making the hypothesis that objects are represented in the scene as clusters of points right above a main plan, approaches such as *Tabletop object detector* [11] and Caron et al. [8] determine the main image plane and search for objects clusters on top of it. Feature extraction is either RGB-based or geometrical. Among RGB features, bag of words of SIFTs or color histograms are commonly used [10]. Deep features [25] have also received a lot of attention recently, but their high dimensionality requires high computation performance to be processed in real time. Geometrical features based on point clouds are either local descriptor, such as SHOT [23], semi-global such as CVFH [2], or global [21], [22].

Ambiguous viewpoints can easily trick single-view recognition approaches. Yet, considering the observation of objects as a sequence of distinctive but known points of view seems to be a natural way to deal with the problem. In that regard, a review of multi-view object representation can be found in [20]. Moreover, a solution inspired by human behavior for learning new unseen objects has been proposed by [7], using key-frames and the rate of matching features with past frames, to overcome ambiguity in facial recognition task. Other approaches also use 3D CAD models [3] and graphs [9] to enhance the internal representation of objects. Last, active approaches aim to drive the robot displacement to improve its perception and reinforce the recognition capability, either based on reinforcement learning [17] or hidden sides estimations [5]. Following the idea of fusing both visual and odometric data to improve recognition, Le Barz et al. proposed an approach to combine visual recognition and odometry to retrieve the most probable localization of a drone in a urban environment [15].

In this paper, we propose an object and object viewpoint recognition system based on both RGB-D representation and odometric data. For that, we build a representation of objects based on views and transitions between them. On the one hand, a database of known objects is stored with different views of the same object and odometric transition between them. On the other hand, the sequence of observation during the recognition stage is modeled as a HMM, and the odometry allows us to retrieve the most probable sequence of observations compared to our database. Our method also

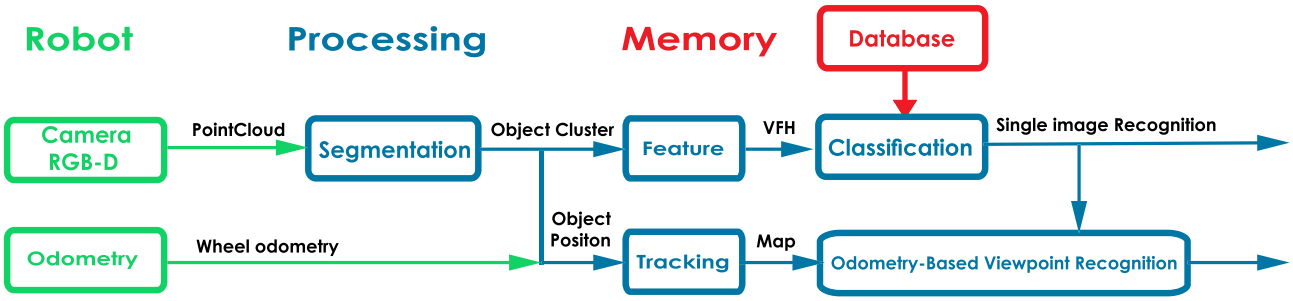

Fig. 1. General achitecture of the system

relies on a segmentation algorithm that isolates objects and a mutli-objects tracker to deal with the case where several objects are present at the same time in the scene.

II. PROPOSED APPROACH

A. Overview

Our approach was designed for robots equipped with odometric and RGB-D sensors. For this system to work correctly, a way to convert the odometry data to robot position and orientation is necessary. This part is not described in this article and the conversion between odometry and robot position is considered as solved in the following explanations.

Informations from odometry and RGB-D units are sent to a processing module that isolates objects from input images, extracts their characteristics through features, and compare them with a reference database stored in memory. When no matching is found between observation and the current database, a new object can be added to the later, increasing the knowledge of the robot about the environment.

The architecture of the system is illustrated in Figure 1 and explains the dependencies between the different processing modules and the information flow through them.

Precisely, the processing units takes as input the point cloud from the RGB-D sensor as well as the odometry measure. The first module consists in a segmentation stage that cleans up the point cloud by isolating objects of the scene into clusters of points. Those clusters are then sent to the feature extraction module to convert them to discriminative description histograms. Simultaneously, a cartography module converts the odometry data and the segmented object image position into the localization of scene elements in the world referential. Finally, the recognition module uses both the object feature and the evolution of those features compared to the displacement of the robot to retrieve the most likely object and view.

B. Object Segmentation

The segmentation algorithm used in this work is the one proposed by Caron et al. [8]. The approach uses the same assumption that objects are lying on planes surfaces that can be found as the major plane of images. Furthermore, on our robot, the camera is always keeping the same orientation towards the floor. In this manner, its plan equation can be estimated once during a calibration step and used during the whole experiment. The segmentation algorithm uses the following steps

0) Calibration to obtain the floor plane equation before experiments, using RANSAC algoritm on an image where floor is the main plane.

Then for each frame of the experiment:

1) Floor subtraction from the obtained equation
2) Points filtering if the distance to the sensor is more than 3 meters
3) Normal surfaces of the remaining points calculation
4) Filtering of big planes orthogonal to the ground. They are very likely to be walls
5) Projection of those points on the floor plane
6) Clustering of the projected points to create object candidates
7) Elimination of clusters that are too close to borders. They might not be objects, and their missing parts are likely to badly influence the recognition.
8) Centroid calculation of each remaining cluster (should be the objects of the image)

Based on this algorithm, we obtain the position of each object in the camera frame, as well as the associated point cloud and normals.

C. Object Representation

The segmentation module provides a point cloud for each isolated object in the observed frame. This raw information is however not enough to efficiently discriminate the observed object and retrieve the associated view in the database. From the point cloud, we then extract two types of information. The first dimension of information relates to its position in the environment. This cartographic data allows the robot to track objects while moving around the scene, consequently, associating multiple views of it. A second dimension that can be translated in terms of visual and geometrical features is related to intrinsic characteristic of the objects capable of differentiating them from one another.

1) Features: In our system, a global descriptor for each view is the easiest and best suited approach, as we need a

way to efficiently compare a large number of view between a database and the current observation. We also restrict our descriptor to geometrical features that we found more discriminative than visual ones.

Among global geometrical features, we selected the Viewpoint Feature Histogram [22] - VFH. This feature captures in a single histogram the geometry of the object by estimating the angular transformation between the normal of each of its points and the standpoint from where it has been viewed. It is therefore discriminative of the object, but also the viewpoint at which it was observed.

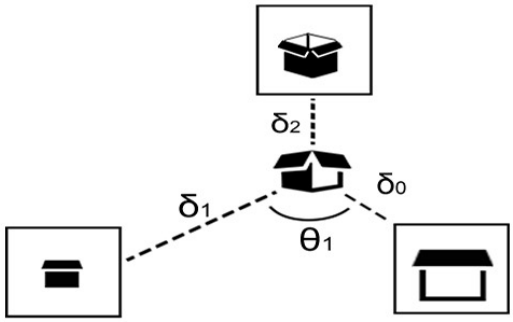

Fig. 2. Multi-view object reprensentation based on a polar aspect graph

2) Multi-view object representation: We define a multi-view representation based on polar aspect-graphs in order to store objects in the memory of the robot. This representation fuses both the intrinsic characteristics of the views of objects as well as the necessary movement to change between them.

The polar referential is centered at the centroid of each object, illustrated in 2. Its nodes represent the object aspect from a certain point of view, translated into a VFH histogram, and transitions from nodes are angular displacements obtained by the robot odometry. The use of VFH instead of viewpoint invariant features is then justified by this type of representation. Indeed, viewpoint information can bias the classification to search for eligible candidates in the database.

D. Object Recognition based on multiple views and their relative positions

Moving the robot around the environment produce a series of object observations. The idea behind our system is to find consistency between this sequence of possibly misrecognized objects given a known displacement. In other words, tracking the positions of objects and the robot provides us an estimation of the physical transition between two successive viewpoints which can reinforce candidates as they are consistent with the movements prediction. To this end, we use a probabilistic model based on hidden Markov models.

1) Viewpoint recognition based on a single observation: Regardless of odometry and cartographic data, a single observation provides a VFH histogram for each segmented object. Determining the viewpoint of the object consists in finding the closest histogram in the database. To this end, the work from Aldoma et al. [1] suggest the use of a K-Nearest Neighbors, based on a chi-squared distance between histograms. The K-NN classifier has the great advantage of having both classification and training stages extremely fast for the intended database size. Thus, the closest viewpoint V from the observation O is retrieved based on the following equation:

$$\begin{cases} V = \underset{v_1 \dots v_k}{\operatorname{argmin}}\ d(O, s_k) \\ d(O, s_k) = \sum_i \frac{\left\| H_O(i) - H_{s_k}(i) \right\|^2}{H_O(i)} \end{cases} \tag{1}$$

with $H_p(i)$ the ith bin of the VFH histogram of a point cloud p and $s_1, \dots s_k$ are the different views available in the database. Moreover, the distance $d(O, s_k)$ between an observation O and a view s_k in the database can easily be turned into a probability $P(O|s_k)$

$$\mathbf{P}(O|s_k) = \frac{d(O, s_k)}{d_{max}} \tag{2}$$

where d_{max} is the maximum distance between two views of the database.

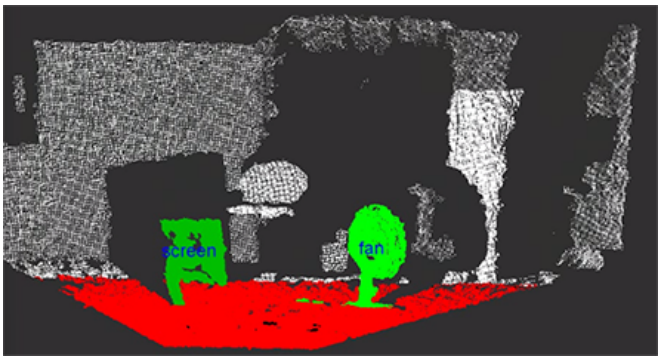

Fig. 3. **Result of object segmentation and identification** The red points represent the floor plane and the ignored white ones exceed the three meters threshold. One could also notice the infra-red shadows created by objects, potentially leading to occlusions

2) Object Tracking and Angular Displacement Estimation: The segmentation algorithm provides us at each new frame a set of point clouds (one for each object) and associated centroid position in the camera sensor reference frame. On the other hand, we receive a signal of the speed of the wheels of the robot that we convert into a displacement estimation in the absolute reference frame. Therefore, we can convert both the position of the robot and point cloud centroids to the absolute reference frame. Nevertheless, imprecision from communication delays between processing modules, segmentation and object centroid calculation call for a filtering method to refine the absolute position of the observed objects. Moreover, because of the segmentation restrictions and constant displacement of the robots, objects are alternatively present and absent from the input image.

For that reason, we rely on a multi-tracker based on Kalman filtering that both refines the positions of each observed centroid and keeps an history of all observations and angular displacement for each tracked object from the beginning of the experiment.

The multi-object tracker is composed with a set of Kalman filters, each of them tracking a single object and being updated as soon as a new observation of the object is found. Briefly, the update rule is done as follows: For each new frame, the centroid of all segmented objects are converted to the absolute reference frame. Then, their positions are compared with the position of the last observation of each available tracker (initially, no tracker is available). If the position of a centroid is close enough to the last observation of a tracker, the tracker is updated using Kalman update equations, and the pointcloud is added to the tracker history. If a centroid does not match with any Kalman position, the centroid is considered as a new object and a dedicated tracker is initialized.

This multi tracker relies on the two asumption that objects of the scene are static (although a dynamic model could be added) and that objects are separated of a minimum distance to avoid confusions between trackers.

Between two consecutive views $k-1$ and k of the same object, an angular displacement is estimated to store the tracked object history with a polar aspect-graph representation (See section II-C.2). The angular displacement is obtained by the following equation

$$\delta_{k-1,k} = atan(p_{k-1} - p_{obj}) - atan(p_k - p_{obj}) \tag{3}$$

with p_{k-1} and p_k the absolute position of the robot for views $k-1$ and k, and p_{obj} the absolute position of the centroid of the tracked object.

3) HMM filtering: Based on the sequence of observations and estimated angular displacement obtained with the odometry, we use Hidden Markov Models to retrieve the closest sequence of views in the database. This sequence not only provides us a reinforced recognition of the object, but also an estimate of the successive views at which the object was seen. The definition of our approach is inspired from [15].

Given a track T associated with k observations $O_1, ..., O_k$ and $k-1$ angular transitions $\delta_{1,2}, ..., \delta_{k-1,k}$ between each of them, we model the viewpoint of the object from the robot sensor tracked by T at time k by a random variable V_k that follows a Markovian process.

Based on the notations used in [18], we define our HMM as $\lambda = \{N, M, \Pi, A, B\}$, with N the number of states, M the number of observations, Π the initial state vector, A the transition probability matrix, and B the observation probability.

In our case, a hidden state is associated with each viewpoints s_i, $0 < i \leq N$ in the database. N is therefore the number of views in the database. As no prior assumption is made about the initial state of the system, we set Π as a vector of length N with equal probabilities.

The state transition probability matrix $A = \{a_{i,j}\}$ is defined as

$$a_{i,j} = \mathbf{P}(V_k = s_i | V_{k-1} = s_j) \tag{4}$$

or in other words, the probability of seeing view s_i at observation k given view s_j at observation $k-1$. This probability depends both on the angular displacement $\delta_{k-1,k}$ between the two observations, and the angular distances $\theta_{i,j}$ between views s_i and s_j in the database. Therefore, the transition probability is modified as each observation k and defined as

$$a^k_{i,j} = \begin{cases} 1, & \text{if } \theta_{i,j} < 2\delta_{k-1,k} \\ \varepsilon_1, & \text{otherwise} \end{cases} \tag{5}$$

A small probability ε_1 is attributed to unlikely transitions between the same object views to reinforce objects recognition rather than pose consistency. Transitions between views of different object are not allowed with zero probability of happening.

Last, the observation probability matrix $B = \{b_j\}$ is also constructed from each new observation. The distance from the K-NN is transformed in a probability distribution of the K neighbors.

$$b^k_j = \begin{cases} \mathbf{P}(O_k | V_k = S_j), & \text{if } s_j \in D_K \\ \varepsilon_2, & \text{otherwise} \end{cases} \tag{6}$$

where D_K is the set of K nearest neighbors and $\mathbf{P}(O_k | V_k = S_j)$ is the converted histogram distance into probability, defined by equation 2. If the state view S_j is not among the K nearest neighbors, a small probability ε_2 is associated instead.

Last, the Viterbi [12] algorithm is used to find the most likely sequence of views $\mathbf{V}$ given the HMM λ, and the sequence of observations $\mathbf{O} = \{O_1, ...O_M\}$.

$$\mathbf{V} = \underset{\mathbf{S}}{\text{argmax}}\ \mathbf{P}(\mathbf{S} | \mathbf{O}, \lambda) \tag{7}$$

The Viterbi path of the track with observations $\mathbf{O}$ then coincides with the sequence $\mathbf{V}$ of recognized view from $\mathbf{O}$. The optimization is solved using the following equations

$$\begin{aligned} V_{1,i} &= b^1_k \cdot \pi_i \\ V_{k,i} &= \max_{j \in \{1,...N\}} \left(b^i_k \cdot a^k_{j,i} \cdot V_{k-1,j} \right) \end{aligned} \tag{8}$$

where π_i is the probability of initially being in ith state (from Π vector), and $V_{k,i}$ is the probability that the most probable sequence based on the k first observations ends up in state s_i. The most probable sequence is then retrieved by taking the sequence of V of successive states used to compute $V_{M,i}$ so that

$$s_i = \underset{s_j, j \in \{1,...N\}}{\text{Argmax}}\ (V_{M,j}) \tag{9}$$

III. Experimental results

The proposed recognition system was deployed in a differential mobile robot, the Wifibot V2, embedded with a RGB-D camera, Asus Xtion Pro Live. The algorithm architecture was implemented over ROS using PCL and OpenNi2 libraries.

In order to validate the approach, the robot was initially taught aspects graphs from chosen objects and two sets of experiments were proposed to analyze the efficiency of the algorithm in realistic scenarios.

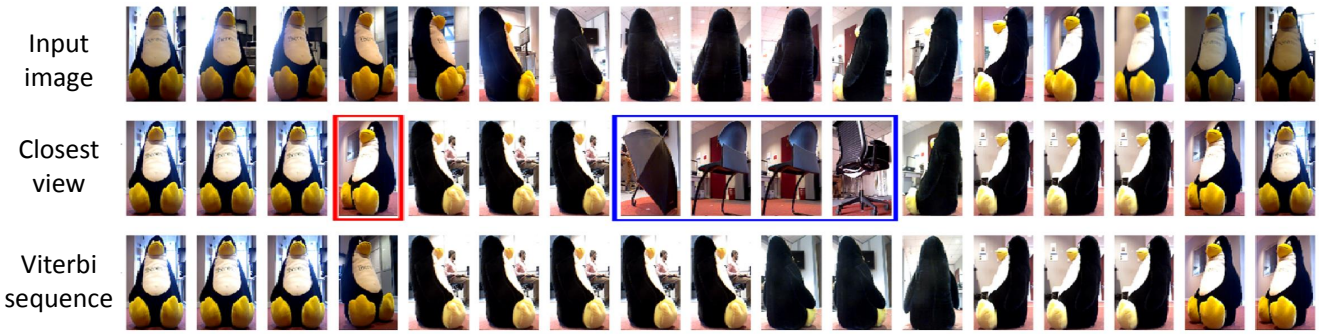

Fig. 4. **Typical experiment** - The HMM refinement makes self correction of irrelevant views estimation, even when objects were badly segmented during the database creation.

A. Object Database

First, twenty objects varying in size and shape (monitor, boxes, chairs, trashcans, people, ...) were selected to compose the knowledge database of the robot. The aspect graphs of each object were obtained with VFH features from *eight* equally distant viewpoints acquired from positioning the robot around the to be learned object 1.5 meters away. This way, the angular position between two consecutive views was easily and accurately determined and stored.

B. Odometry Contribution

The first experiment consist in a comparison between the performance in recognition between single-view and multi-view recognition techniques. In other words, this comparison attest whether the proposed architecture is interesting or not, since having the same or worst performance by the cost of adding a complex post-processing and tracking modules is not interesting.

Technically, each object was partially circled by the robot from at least four random initial positions at speed of $0.35 \pm 0.1 m/s$. The robot recorded for each run between 5 and 30 frames at different viewpoints of the object, and tried to recognize them using both recognition algorithms.

A typical experiment is illustrated in Figure 4. The first row represents the image sequence segmented by the robot for a given object at each new frame. This is the object to be recognized. The second line, is the viewpoint recognition based on a single image, without using any odometric information. The only measure used to recognize the view is provided by the distance to the nearest neighbor in the database (see section II-D.1). Interesting is to notice that the rotation invariance of the VFH descriptor actually tricks the view estimation by taking the symmetrical correspondent in the first red square. In addition, the back of the penguin is a large and almost flat surface. It is, therefore, considered as background wall by the segmentation module and almost entirely removed. The resulting point cloud is insufficient to accurately characterize the view, thus leading to wrong view detection in the blue squares. The last row represent the final output of the Viterbi algorithm. As expected, the combination of VFH-based view estimation and odometry is able to retrieve the correct sequence of views.

The experiment was then numerically evaluated by comparing the performance of the two approaches as the number of views was increased, displayed in Figure 5. Dashed curves represent object and view recognition on a single image basis. Solid ones are obtained based on multi-view estimation. When the number of observation is low, the mono-view recognition tends to be slightly better than multi-view. The gap between mono and multi-view recognition increases with the number of observations. With 25 or more observations, the multi-view recognition outperforms the mono-view by 30% for object recognition and 40% for view estimation. Last to make sure that odometry has an impact in the object recognition process, we compare the results obtained from the Viterbi sequence ('*multi+odom obj*') with the recognition rate when using only the average probabilities provided by equation 2 over the sequence ('*multi obj*'). In this case, the recognition rate gap is around 7%.

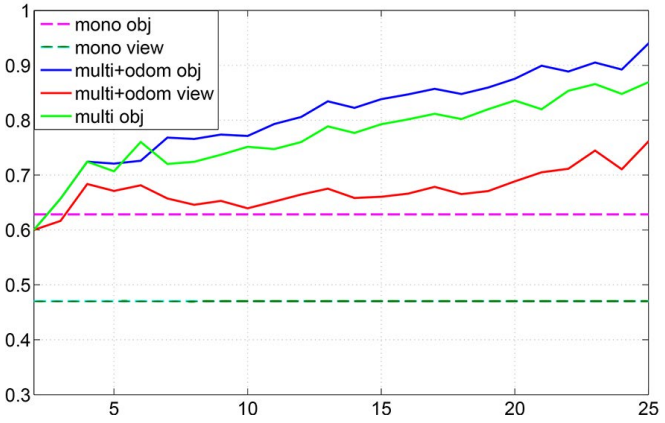

Fig. 5. **Numerical results evaluation** - Evolution of the correct recognition rate versus the number of views. Both for objects and view recognition, the multi-view approach outperforms the mono-view approach.

C. Multi-object recognition

In a second evaluation, concurrently with the recognition efficiency, the multi-tracking capabilities were put into test. Tracking objects correctly has a critical impact on the overall performance of the system.

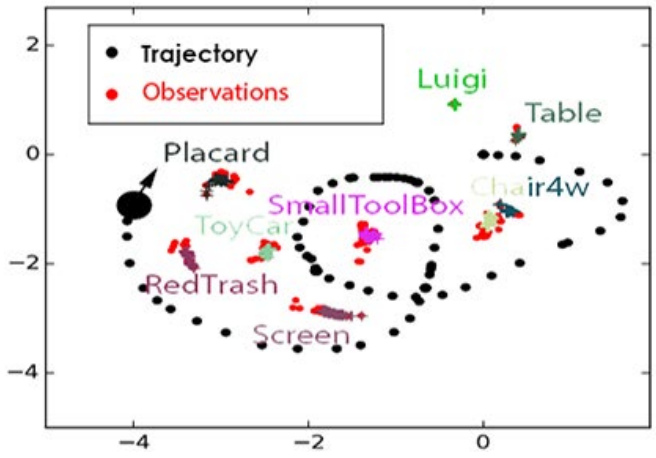

Fig. 6. **Mutli-objects tracking results** - Red dots corresponds to segmented objects centroids. Each color cross stands for the position estimation from a tracker. Each filter is associated with a unique color. Theoretically, a single filter should be associated with a single object. Black dots represent the trajectory of the robot during the experiment.

The experiment consists in moving the robot around a room full of objects, making sure to visualize them from different perspectives. A typical view from the robot as well as associated segmentation result is presented in Figure 3. This scenario is a much more complex case than the first one as moving in a room with many elements generates occlusions, thus leading to incomplete point clouds and biased feature descriptors. An simple evaluation of the multi tracking is displayed in Figure 6. Thee possible scenarios can leard to tracking failure. First, the object is never segmented and no track is open for it. Second, more than one track are opened for the same object. This may happen if an object is segmented in two different clusters or it is not seen for a long time and its estimated position error is too high . Third, two objects are merged in the same track. This may happen if objects are too close from each other. In the case of Figure 6, the object 'Chair4W' has two tracks for the same object. On the other hand, Figure 7 provides another sequence example on which two different tracks were fused. In this case, the person and the fan were too close from each other, and a track mixing both entries was created. The result is the identification of the 'fan' in a track that is mainly composed with 'person' observations.

In the sequence presented in Figure 7, Four of the five objects present in the scene were correctly recognized. As explained earlier, the only object recognition mistake is because the first object, a person, was too close to the fan and were considered as one single object by the tracker.

21/36

IV. Conclusion

The proposed method focuses on integrating odometry information to boost recognition efficiency, based on the intuition that different perspectives of the object and analysis of transitions between them provides unexplored information. The results from practical experiments shows that the

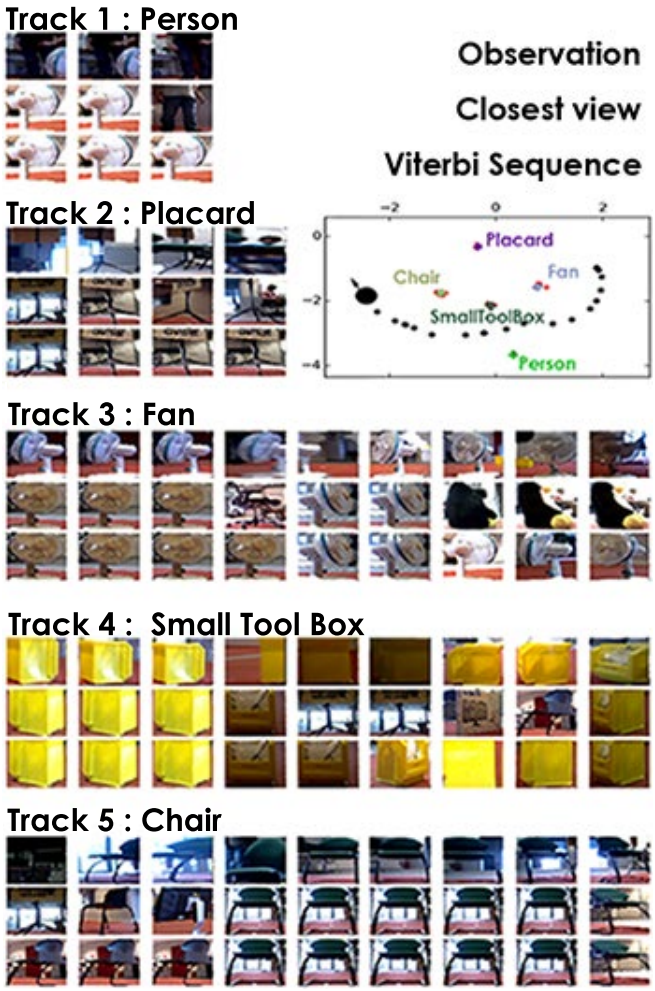

Fig. 7. **Example of multi-view recognition in a scene with five objects** - For this run, odometry-based recognition got 80% of the objects and 42 % of the views right, the mono-view got 34% for both, and the multi got 80% right for object classification

odometry-based algorithm outperforms the classical one in both object and viewpoint recognition, strengthening the interest in a more sophisticated pipeline. Moreover, the approach has the advantage of not being platform specific, which means it could be implemented in any mobile system equipped with a RGB-D sensor and capable of measuring its position. However, the algorithm is quite sensitive to bad segmentation, specially if a object is divided into two disjoint clusters. Errors due to occlusion could be reduced using semi-global descriptors and a more exigent segmentation.

Future work will consist in recognizing moving objects, where the recognition feeds the tracking module with the moving dynamics of the target, and including object transformation by allowing state transitions to views of other objects [1]. A more immediate challenge is to create the object database on the fly once a new observation is not encountered on it, according to a distance threshold.

[1]i.e. human moving towards a chair and sitting down, creating a new object human+chair

REFERENCES

[1] A. Aldoma, Zoltan-Csaba Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R.B. Rusu, S. Gedikli, and M. Vincze. Tutorial: Point cloud library: Three-dimensional object recognition and 6 dof pose estimation. *Robotics Automation Magazine, IEEE*, 19(3):80–91, Sept 2012.

[2] Aitor Aldoma, Federico Tombari, Radu Bogdan Rusu, and Markus Vincze. *OUR-CVFH–Oriented, Unique and Repeatable Clustered Viewpoint Feature Histogram for Object Recognition and 6DOF Pose Estimation*. Springer, 2012.

[3] Aitor Aldoma, Markus Vincze, Nico Blodow, David Gossow, Suat Gedikli, Radu Bogdan Rusu, and Gary Bradski. Cad-model recognition and 6dof pose estimation using 3d cues. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 585–592. IEEE, 2011.

[4] 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.

[5] Joseph E Banta, Laurana M Wong, Christophe Dumont, Mongi Abidi, et al. A next-best-view system for autonomous 3-d object reconstruction. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 30(5):589–598, 2000.

[6] M Bjorkman and Danica Kragic. Combination of foveal and peripheral vision for object recognition and pose estimation. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 5, pages 5135–5140. IEEE, 2004.

[7] Heinrich H Bulthoff, Christian Wallraven, and Arnulf Graf. View-based dynamic object recognition based on human perception. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 3, pages 768–776. IEEE, 2002.

[8] 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.

[9] Indranil Chakravarty and Herbert Freeman. Characteristic views as a basis for three-dimensional object recognition. In *1982 Technical Symposium East*, pages 37–45. International Society for Optics and Photonics, 1982.

[10] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague, 2004.

[11] Ros documentation. Tabletop object detector.

[12] G David Forney Jr. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[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] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[15] Cédric Le Barz, Nicolas Thome, Matthieu Cord, Stéphane Herbin, and Martial Sanfourche. Global robot ego-localization combining image retrieval and hmm-based filtering. In *6th Workshop on Planning, Perception and Navigation for Intelligent Vehicles*, pages 6–p, 2014.

[16] 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.

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

[18] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[19] Sanjay Dhar Roy, Santanu Chaudhury, and Sean Banerjee. Isolated 3d object recognition through next view planning. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 30(1):67–76, 2000.

[20] Sumantra Dutta Roy, Santanu Chaudhury, and Subhashis Banerjee. Active recognition through next view planning: a survey. *Pattern Recognition*, 37(3):429–446, 2004.

[21] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3212–3217. IEEE, 2009.

[22] Radu Bogdan Rusu, Gary Bradski, Romain Thibaux, and John Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2155–2162. IEEE, 2010.

[23] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In *Computer Vision–ECCV 2010*, pages 356–369. Springer, 2010.

[24] Chieh-Chih Wang, Charles Thorpe, Sebastian Thrun, Martial Hebert, and Hugh Durrant-Whyte. Simultaneous localization, mapping and moving object tracking. *The International Journal of Robotics Research*, 26(9):889–916, 2007.

[25] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems*, pages 487–495, 2014.