A Deep Learning Based Behavioral Approach to Indoor Autonomous Navigation

G. Sepulveda, J. C. Niebles, and A. Soto

Abstract— We present a semantically rich graph representation for indoor robotic navigation. Our graph representation encodes: semantic locations such as offices or corridors as nodes, and navigational behaviors such as enter office or cross a corridor as edges. In particular, our navigational behaviors operate directly from visual inputs to produce motor controls and are implemented with deep learning architectures. This enables the robot to avoid explicit computation of its precise location or the geometry of the environment, and enables navigation at a higher level of semantic abstraction. We evaluate the effectiveness of our representation by simulating navigation tasks in a large number of virtual environments. Our results show that using a simple sets of perceptual and navigational behaviors, the proposed approach can successfully guide the way of the robot as it completes navigational missions such as going to a specific office. Furthermore, our implementation shows to be effective to control the selection and switching of behaviors.

I. INTRODUCTION

Currently, the autonomous navigation system of most mobile robots relies on a fine-grained geometric world representation, e.g., a metric map [1]. As a main drawback, this type of representation exhibits a strong reliance on low level structural information, such as specific geometric configurations of the environment [2] or low level visual cues [3][4], limiting its generality and robustness. This is particularly relevant in GPS-denied environments, mainly indoor spaces, where tasks such as robot localization and planning/exploration, strongly depend on the underlying representation of the environment. As a further disadvantage, the use of a low level representation limits the ability of the robot to interact with human users, who usually understand the environment, and refer to it, at a higher level of abstraction. Recently, the so-called semantic mapping techniques emerge as a step forward to improve the previous limitations [5]. These approaches, however, are usually built on top of low-level geometric representations, inheriting most of their problems [6].

At a more fundamental level, we believe that current representations for robot navigation do not take full advantage of the rich semantic structure behind man-made environments. In particular, most man-made environments are designed to facilitate human navigation. Consequently, they are mainly composed of navigational structures, such as sideways, corridors, or stairs, that in turn are intended to connect meaningful neighboring places, such as houses, rooms, or halls. We hypothesize that, by providing robots with suitable abilities to understand the world at this semantic

G. Sepulveda and A. Soto, Dept. of Computer Science, Pontificia Universidad Catolica de Chile, (gsepulveda@uc.cl, asoto@ing.puc.cl)
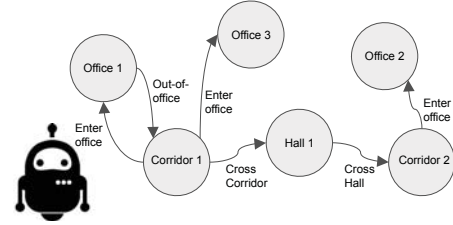J. C. Niebles Dept. of Computer Science, Stanford University, (jniebles@stanford.edu)

Fig. 1: Proposed representation for robot navigation. Nodes represent perceptual behaviors used by the robot to recognize specific semantic places, such as *Office-1* or *Corridor-2*. Edges represent navigational behaviors used by the robot to move between relevant places, such as *leave-an-office* or *cross-a-corridor*.

level, it is possible to equip them with navigational systems that largely exceed the generality and robustness of current approaches.

In this work, we revisit early approaches based on a graph-based cognitive map of the environment [7]. In particular, for the case of an office building, we propose a graph representation similar to the one depicted in Figure 1. In this graph, nodes correspond to *perceptual behaviors* which are intended to recognize specific instances of high level semantic structures, such as particular offices, corridors, or halls. Edges in the graph correspond to *navigational behaviors*, such as cross-a-corridor, leave-an-office, or enter-a-room, which are intended to lead the robot as it navigates between semantic places.

Departing from current SLAM approaches [5][6], the proposed representation questions the need to keep an explicit estimation of metric or low-level geometrical information about the environment. To illustrate this idea, consider the case of a robot that needs to cross a corridor. To achieve this goal, the robot does not need to know its precise position within the corridor. Instead, it can succeed by just strictly following the corridor until reaching its end. In this sense, we advocate for a representation based on a perception for action paradigm, where perception is an attentive modality that focuses on the basic needs to achieve the current robot goals [8].

The proposed scheme is reminiscent of early behavioral based approaches for autonomous robot navigation [9][10]. These early attempts rely on low level sensory information or hand-crafted visual features that, at the time, lacked of sufficient robustness to deal with the complexities of natural environments. Fortunately, current deep learning (DL) techniques open a new opportunity to revisit these ideas. In effect, DL offers the possibility to provide a robot with learning skills to acquire suitable behaviors by directly experimenting with a real [11] or a simulated environment [12].

As a proof of concept, in this work we experiment using virtual environments that simulate office building spaces.

This facilitates the acquisition of training data as well as the evaluation of different aspects of the proposed approach. Our main goal is to demonstrate how a robot can navigate in these simplified spaces using a set of predefined perceptual and navigational behaviors. Our implementation follows recent work on visual problem solving using perceptual modules [13] [14] but, in contrast to those works, our planning problem involves a temporal dimension that, at execution time, implies the determination of suitable switching times between behaviors. Specifically, during an initial exploration phase, the robot builds a representation of the environment by recording the set of behaviors that are being activated. Afterwards, the robot uses this representation to solve and execute planning problems, such as going from semantic place A to semantic place B. At planning time, this implies selecting and sorting a suitable set of behaviors to reach the intended goal. At execution time, this implies the correct identification of places and transition times between the activation of navigational behaviors.

As a distinguishable feature, this planning scheme resembles the semantic representation used by a human to guide a stranger in a new environment. Indeed, when presenting a new environment, the usual directions provided by a human are related to the activation of perceptual and navigational behaviors, such as *"cross the hall, take the corridor to the right, and walk until you find a large door"*. In this sense, during navigation, the activation of the intended actions (navigational behaviors) is usually associated to the identification of specific perceptual cues (perceptual behaviors). These are the guiding observations behind our proposed approach. In particular, we make the following main contributions:

- A new approach for indoor robot navigation that offers two main advantages: i) A navigation scheme centered in action selection that is less prompt to localization errors, and ii) A semantically rich and compact representation that facilitates path planning and interaction with human users.
- A working system that uses a virtual environment to show the viability of the proposed approach.
- An open source implementation that will be available online to foster further research in this area.

II. RELATED WORK

There is an extensive literature discussing the problem of autonomous robot navigation, we refer the reader to [5] for a recent review. Here, we focus our discussion on indoor environments, where the robot does not have access to a global positioning system (GPS). Similarly, in terms of the abundant literature about DL models [15], we focus our discussion on applications related to visual indoor navigation.

Inspired by [16], behavioral approaches to robot navigation gain high popularity during the 80s and 90s. In particular, the subsumption architecture becomes one of the best known approaches [9]. Under this scheme, behaviors are implemented as finite state machines that are combined through mutual suppression mechanisms (subsumption). As a key underlying principle, subsumption states that the world is its own best representation, therefore, there is not need to keep an explicit internal representation of the environment. Instead, a robot can directly uses its current estimation of the state of the world to trigger the opportunistic activation of suitable behaviors. Subsumption based robot navigation shows remarkable success to emulate reactive behaviors observed in biological systems, such as insects [17]. It also achieves some degree of success to perform simple tasks in natural scenarios, mainly office buildings [10].

In contrast to subsumption approaches, later research has stressed the need to keep an internal representation of the world in the form of a map of the environment. In particular, the mapping and localization problems have been usually handled in conjunction, leading to the Simultaneous Localization and Mapping or SLAM problem [1]. Initial SLAM approaches are based on the Kalman Filter (KF) and its extensions [18]. Limitations of the Gaussian assumption in KF techniques lead to non-parametric approaches based on the Particle Filter [19]. Later, limitations of non-parametric approaches to scale to large environments lead to Graph-Based methods [20]. Under this scheme, the challenge is to find a configuration of the nodes in the graph that is maximally consistent with the observations of the environment, which implies solving a complex optimization problem. Several works have proposed highly efficient solutions [21][22] that are able to operate in real time for large environments ($> 10^8$ landmarks) [23][24]. While SLAM approaches make intensive use of range and odometry sensors, advances in computer vision techniques facilitate the development of the so-called Visual-SLAM (V-SLAM) approach based on visual sensors [4]. As an example, [25] presents FAB-MAP, a SLAM approach based on the detection of 2D visual keypoints. Extensions using RGB-D sensors have also been proposed [26].

The previous SLAM techniques rely on low level structural information which limits their generality and robustness. Moreover, they lack of a suitable semantic representation that facilitates the interaction with human users. This has motivated recent extensions that incorporate high level semantic information [5], such as the ability to recognize objects [27][6], scenes [28], or navigational structures [29]. As an example, [6] proposes a Graph-Based SLAM approach that includes the detection of semantic objects, such as chairs and doors. As expected, results in [6] indicate that the understanding of the environment at a higher level of abstraction helps to increase navigational robustness.

Closer to our ideas, there are recent works on robot navigation using a DL approach, mainly deep reinforcement learning (DRL) techniques. [30] presents the so-called Cognitive Mapper and Planner (CMP), a system that integrates mapping and planning using a joint learning scheme. Given visual and self-motion information, this system learns a policy to achieve generic goals, such as go to a target location or find an object of interest. Similarly, [12] presents a system that learns a policy to move a robot to a given view point position of the environment. In other words, the goal is given by an image of the target location. Interestingly, both works, [30] and [12], use virtual environments to obtain a first navigational model that is then refined using real data. In contrast to our work, [30] and [12] do not learn an explicit representation of the environment and they are not based on a behavioral approach.

In terms of works that use DL to learn an explicit map of the environment, [31] presents the so-called Neural Map, a system that keeps a 2D representation of the environment

similar to a 2D occupancy grid [32]. As a relevant feature, instead of storing binary information in each map cell (busy/empty), they store a feature vector provided by a DL model. In a related approach, [33] presents a navigation approach based on a temporal extension of Memory Networks [34]. As a result, they are able to improve navigational performance with respect to pure reactive approaches that do not consider an explicit external memory. [35] proposes a DRL approach that improves robot navigation by considering auxiliary tasks, such as depth prediction and loop closure classification. As a result, they are able to improve navigational performance with respect to approaches that only consider the navigational task. In contrast to our approach, none of the works above use an internal representation based on navigational behaviors.

In the biological side, the use of an internal representation of the world has been supported by highly influential studies that demonstrate the use of an explicit representation of the world by the navigational systems of rodents and humans [36][37]. Here, we also advocate for the use of an explicit internal representation of the world. As a main novelty, we propose an approach that deeply connects this representation to the execution of goal oriented behaviors.

III. PROPOSED METHOD

Our key innovation is the use of a representation of the environment given by a set of perceptual and navigational behaviors. Furthermore, similarly to [11], these behaviors are robustly learned using deep learning techniques, specifically supervised imitation learning. This scheme marks a departure from current state-of-the-art SLAM techniques that mainly rely on metric and landmark-based representations of the environment.

Figure 2 shows a hypothetical case that helps to illustrate our representation as well as to highlight its main differences with respect to current approaches for indoor robot navigation. Figure 2-a shows an example of a typical map representation used by current robots to navigate in an office environment. This consists of a metric map (grey structures) augmented with information about relevant visual landmarks (circles with a black dot in the center), while the metric representation encodes the configuration of relevant structures and objects, the landmark-based topological representation increases localization robustness by encoding the appearance and relative position of specific visual landmarks. As a relevant fact, the key component of this hybrid representation is the encoding of geometrical information about the environment. Figure 2-b represents the same environment, but using the labels of meaningful semantic locations, such as offices (O1, O2, ...), halls (H1), and corridors (C_{1l}, ...). Using these labels, Figure 2-c shows an excerpt of an instantiation of the proposed graph representation for the office environment considered in a) and b). As shown in Figure 2-d, this graph can also be represented as a set of navigational rules or triplets that take the form: $\langle place | \textbf{behavior} | place \rangle$[1]. As an example, office O_1 is connected by a navigational behavior *out-of-office-right* (**oor**) to the right direction of corridor C_1 (C_{1r}), leading to the triplet: $\langle O_1 | \textbf{oor} | C_{1r} \rangle$. Similarly, corridor C_1 direction

(a)   (b)

(c)

Map Graph
$\langle place | \textbf{behavior} | place \rangle$
$\langle O1 | \textbf{oor} | C1r \rangle$
$\langle O1 | \textbf{ool} | C1l \rangle$
$\langle O1 | \textbf{ooc} | EO3 \rangle$
$\langle EO3 | \textbf{ior} | O3 \rangle$
$\langle C1r | \textbf{cf} | H1 \rangle$
...
$\langle O16 | \textbf{oor} | C4r \rangle$
$\langle O16 | \textbf{ooc} | EO14 \rangle$
(d)

Plan:
S:O1 G:O8
$\langle O1 | \textbf{oor} | C1r \rangle$
$\langle C1r | \textbf{cf} | H1 \rangle$
$\langle H1 | \textbf{chl} | C2r \rangle$
$\langle C2r | \textbf{cf} | EO8 \rangle$
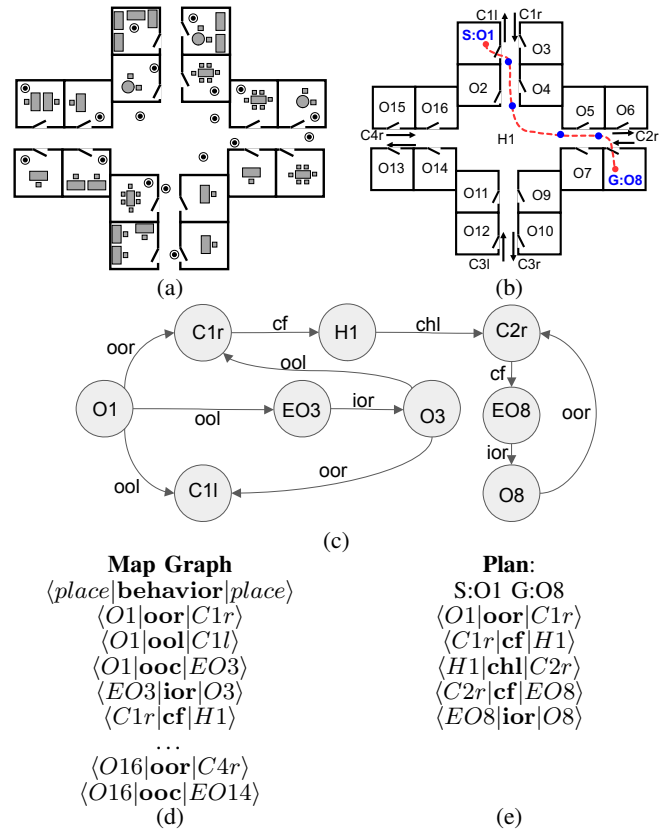$\langle EO8 | \textbf{ior} | O8 \rangle$
(e)

Fig. 2: (a) A typical geometric representation used to implement robot navigation in an office building enviroment. This consists of a metric map (grey structures) augmented with information about relevant visual landmarks (circles with a dot in the center). (b) Instead, we use a graph-based representation derived from the identification of key semantic locations in the environment, such as offices (O1, O2, ...), halls (H1), and corridors (C_{1l}, ...). (c) A partial graph for the environment depicted in b) (see Appendix for notation details). (d) This graph can be represented as a set of triplets of the form $\langle place | \textbf{behavior} | place \rangle$. (e) To navigate from O1 to O8 (as depicted in red in (b)), one can derive a behavioral plan by following appropriate links (triplets) in the graph.

right (C_{1r}) is connected by a navigational behavior *corridor-follow* (**cf**) to hall H_1, leading to the triplet: $\langle C_{1r} | \textbf{cf} | H_1 \rangle$. Notice that given the difference between traversing a corridor through the left or right direction, we introduce a different place label to distinguish these cases.

As a relevant observation, in contrast to metric and landmark-based approaches that focus on encoding geometry, the key component of the proposed representation is the encoding of semantic information about the environment. Where we understand semantic as a set of high-level skills, or behaviors, that allow a robot to successfully navigate in a man-made environment. As a relevant advantage, the proposed representation facilitates the use of planning techniques. As an example, Figure 2-e illustrates a valid plan to go from office O_1 to office O_8, which can be expressed as: Leave office O_1 and take corridor C_1 direction right, $\langle O_1 | \textbf{oor} | C_{1r} \rangle$. Follow corridor C_{1r} until reaching hall H_1, $\langle C_{1r} | \textbf{c}_\textbf{f} | H_1 \rangle$. Then, cross hall H_1 and take to the left to joint corridor C_{2r}, $\langle H_1 | \textbf{chl} | C_{2r} \rangle$. Next, follow corridor C_{2r} until the entrance of office O_8, $\langle C_{2r} | \textbf{cf} | EO8 \rangle$. Finally, enter office O_8 on the right, $\langle EO8 | \textbf{ior} | O_8 \rangle$. The example also

---

[1] See Appendix for notation details.

highlights a relevant additional advantage of the proposed approach, that is, its affinity with the way humans navigate in indoor environments, a property that can largely facilitate the interaction of robots with human users.

The selection and robust implementation of suitable navigational and perceptual behaviors is the key element behind the proposed approach. Indeed, the viability of the proposed approach resides on being able to robustly implement behaviors such as mastering how to leave an office or how to traverse a corridor. As we describe next, DL techniques allow us to acquire such capabilities using training data.

A. Navigational Behaviors.

Following [38] and [11], we use supervised imitation learning (SIL) as our primary strategy to implement the intended navigational behaviors. As a main advantage, SIL takes advantage of a direct observation of how an expert solve similar problems, avoiding a brute force or blind exploration of the state space [39]. Similarly to [11], we implement a SIL scheme based on deep Convolutional Neural Networks (CNNs) and visual information.

As a relevant requirement, the implementation of DL models needs large amounts of training data, in our case, direct observation of expert behaviors. Several strategies can be used to collect this data. As an example, [11] and [40] directly gather training data by recording a first person camera view along with control actions while a human executes a specific behavior. As an alternative, it is possible to gather training data by registering the behavior of a virtual agent while it navigates in 3D reconstructions of natural environments, such as [41], or in photorealistic renderings of virtual environments, such as [12]. As a proof of concept, in this work, we use the virtual environments provided by the research framework DeepMind Lab (DML) [42]. DML provides 3D virtual environments that are built on top of a game engine. In particular, this framework provides a set of tools to collect first-person-view images and control actions while an agent traverses a simulated environment. Figure 3 shows images that illustrate the 3D environments provided by DML.

To generate data corresponding to the navigational behavior of an expert, we augment the DML framework with two main tools. First, we implement a tool that generates random 2D floor plans that simulate office buildings. Specifically, these floor plans are composed of a random arrangement of structural components, mainly office rooms, halls, and corridors. Figure 4 shows examples of configurations generated by this tool. Additionally, using a 2D floor map as input, we implement a second tool that interacts with DML to generate a 3D rendering of the floor map, leading to simulated environments as the ones shown in Figure 3.

Using the previous tools we are able to generate the expert behaviors. Specifically, using a 2D floor plan, we exploit the optimality of traditional path planning techniques [43], to simulate the route that an expert would follow when executing each of the intended behaviors (ex. *entering-an-office*, *corridor-follow*, etc). After calculating an optimal path, a virtual agent executes the route in the virtual world, recording at the same time training data corresponding to visual images and control actions faced by the expert. Additionally, following [40] we account for perturbations with

Fig. 3: Examples of the virtual environments generated by the DML framework [42].

Fig. 4: Examples of office building floor maps randomly generated to gather training data to test the proposed approach. Given a random start location S, a traditional path planner (ex. RRT* [44]) is used to simulate the path that an expert will follow to reach a goal location G.

respect to the expert routes by adding jitter to the trajectory of the virtual agent (views in a $+5°$ and $-5°$ orientations). The previous scheme provides large sources of training data that can be used to learn the intended behaviors.

To obtain the plan to navigate from a start (S) to a goal (G) location, we use the implementation of the planner RRT* [44] provided by OMPL [45]. Figure 4 shows examples of random starts (S) and goal locations (G) as well as the path generated by RRT*. It is important to mention that to generate a path that favors a route that follows the center of the corridor, we use a C-space configuration to increase the thickness of the walls in the map [43]. Furthermore, to closely resemble the path used by a human, we use a cubic spline to smooth the resulting path provided by the planner.

B. DL Models.

In our implementation, we consider two types of behaviors i) *Reactive* and ii) *Memory-based*.
i) Reactive behaviors: correspond to behaviors that implement a direct mapping from sensing to action, i.e., they do not use an explicit internal representation of the world. In our case, this is a direct mapping from visual input to robot motor control. We envision these behaviors as highly general. As an example, a corridor following behavior should be able to successfully operate in a wide variety of indoor environments. As a relevant feature, our navigational behaviors are not purely reactive, but they also incorporate an intended goal. As an example, a navigational behavior to cross a hall also takes into consideration the direction that the robot will take after leaving the hall.
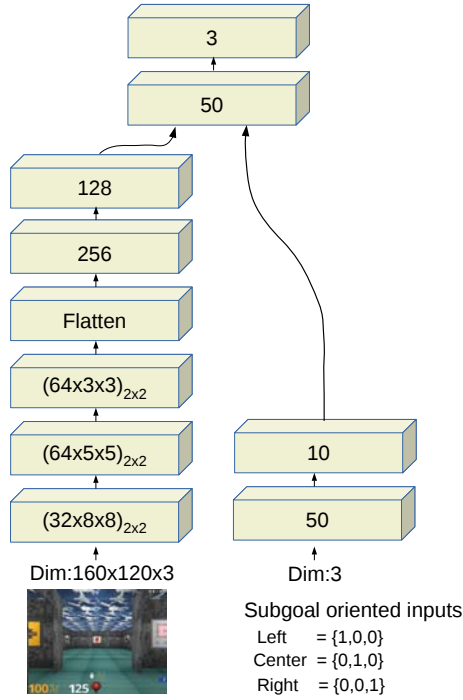
Fig. 5: We implement *reactive behaviors* by mapping directly from visual observations to actions using Convolutional Neural Networks (CNNs). Multiple behaviors may be implemented with a single CNN by using an additional subgoal oriented input. As an example the *out-of-office* behavior can be turned into *out-of-office-right* or *out-of-office-left* depending on the subgoal oriented input.

Figure 5 shows the CNN used to learn the reactive behaviors. The input is given by the current view of the robot and a one-hot 3D vector that provides additional information about subgoals related to the current execution of the behavior, such as leaving a hall using the exit of the left. Given that our robot moves in a flat floor, the output of the CNN is given by a 3D vector indicating the desired translational (x, y) and rotational (θ) velocities. These outputs are given in robot world coordinates (Y axis perpendicular to the robot motion direction).

ii) Memory-based behaviors: correspond to behaviors that use an explicit internal representation, or memory, to recognize previously visited places. In contrast to our reactive behaviors which learn navigational skills that are valid for different environments, our memory-based behaviors encapsulates specific knowledge that is only valid for a particular environment. As an example, a memory-based behavior will have the ability to recognize a specific office or hall. In particular, memory-based behaviors play a relevant role to signal transitions between reactive behaviors. As an example, while following a corridor, the detection of a particular office entrance will signal the switching between a *corridor-follow* and *input-office* behaviors.

Figure 6 illustrates our method to implement a memory-based behavior that allows to identify specific places of the environment. For simplicity, when we generate each environment we associate a specific visual landmark to each meaningful place. Therefore, place recognition is reduced to landmark detection, where these landmarks are given by pictures hanging from the walls (see figures 3 and 7).

Specifically, let's $LM = \{LM_1, \ldots, LM_n\}$ be the set of n known visual landmarks. During training, for each landmark LM_i we obtain a set of m visual descriptors $\widehat{LM}_{ij}$, $j \in [1, \ldots, m]$, by applying a pre-trained CNN to a set of m images that are known to contain landmark LM_i. Each visual descriptor is given by the last pooling layer of the pre-trained VGG-Net model [46], which provides a feature map corresponding to 7x7 image regions. During training, we only consider the image region that most overlap with the known position of the landmark. Afterwards, we apply to the resulting m descriptors $\widehat{LM}_{ij}$ a trainable bilinear function $F(\cdot)$ that provides embbedings $\overline{LM}_{ij}$. We store these embeddins as keys of nxm memory locations. The addressable value of each of these memory locations corresponds to a place-ID PL_i, indicating the place associated to the respective landmark (each landmark appears only once in the environment). Additionally, we also consider a default place-ID PL_{unk} to handle cases where the input image does not contain a valid visual landmark. Consequently, our method for place recognition resembles the association mechanism used by key-value memory networks [34].

At detection time, we apply to the current view of the robot Im, the same pretrained VGG-Net model described above, obtaining a set of 7x7 descriptors. Afterwards, we apply to each descriptor a trainable bilinear function $W(\cdot)$ obtaining encodings Im_l, $l \in [1, \ldots, 7x7]$, that we use to address the memory locations with information about places. In particular, we use cosine distance to score the similarity between each image region descriptor and the encoding of the visual landmarks stored in memory. Using the resulting scores, we estimate a probability $p(PL_i)$ of being at each possible place PL_i. Specifically, we calculate $p(PL_i)$ as follows:

$$p(PL_i) = \sigma(\alpha_i) \qquad (1)$$

where:

$$\alpha_i = \sum_{l=1}^{7x7} \sum_{j=1}^{m} < Im_l, LM_{i,j} > \quad \text{and} \quad \sigma(\alpha_i) = \frac{e^{\alpha_i}}{\sum_i e^{\alpha_i} + e^{\alpha_{unk}}}$$

$< \cdot, \cdot >$ denotes cosine distance, $\sigma(\cdot)$ is a softmax function, and α_{unk} corresponds to a constant coefficient associated to the default place-ID PL_{unk}. This coefficient can be considered as a threshold over the probability of accepting a valid landmark detection. After training, we replace the softmax function in Eq. 1 for a hard maximum operation, reporting as detected place PL_i associated to the coefficient α_i with highest value.

IV. Experiments

As we mention, we test the proposed approach using simulated environments that resemble the configuration of office building spaces. As illustrated in Figure 4, our current implementation considers 3 types of office building spaces: offices, corridors, and halls. In terms of behaviors, while it is possible to explore automatic techniques to identify them [14][47], for simplicity, we manually preselect a set of behaviors that can meet the navigational requirements of the intended environment. Specifically, we implement the following behaviors:
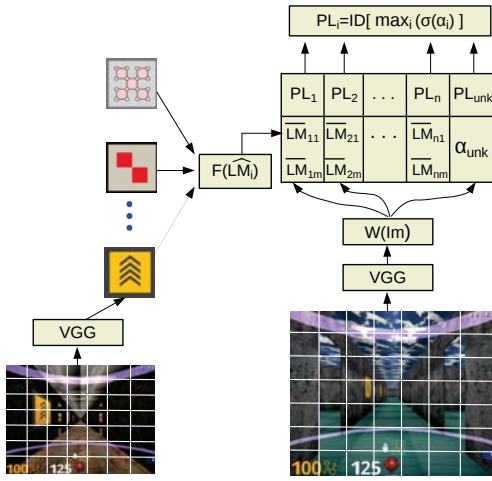
Fig. 6: An example of a memory-based behavior. In particular, this behavior uses a pretrained CNN and a memory module to identify environment places associated to the detection of specific visual landmarks, see main text for detail.

Fig. 7: Sequential set of frames corresponding to samples of a training instance of behavior cross-hall. From left to right, the frames illustrate the robot entering, then crossing, and finally leaving the hall using the right side direction.

- Corridor-Follow (**cf**): a reactive navigational behavior that leads the way of the robot as it navigates through a corridor. The input to this behavior is the current view of the robot (RGB image).
- Out-Office (**oo**): a reactive navigational behavior that leads the way of the robot as it leaves an office. The input to this behavior is the current view of the robot and a 3D vector encoding the direction that the robot should take after leaving the office: *right* or *left*.
- Enter-Office (**io**): a reactive navigational behavior that leads the way of the robot as it navigates through a corridor and enters to the next office on its way. The input to this behavior is the current view of the robot and a 3D vector encoding the direction that the robot should take to enter the next office: *right* or *left*.
- Cross-Hall (**ch**): a reactive navigational behavior that leads the way of the robot as it crosses a hall. The input to this behavior is the current view of the robot and a 3D vector encoding the direction that the robot should follow to leave the corridor: *right*, *left*, or *center*.
- Change-Corridor (**cc**): a reactive navigational behavior that leads the way of the robot as it faces an intersection to follow a new corridor. The input to this behavior is the current view of the robot and a 3D vector encoding the direction of the next corridor: *right*, *left*, or *center*.
- Place Detection (**pd**): a perceptual behavior that identifies the type of place where the robot is currently located. Possible categories are: office, corridor, hall.
- Landmark-based Place Detection (**lmpd**): a memory based perceptual behavior that identifies places using the detection of specific LMs in the environment. Possible categories are a set of $n = 80$ available visual LMs, as described in Section III-B.

A. Behavior training

Input images consist of 160x120 pixels. We test using RGB and gray level images (BW), both schemes converge to suitable models, but BW achieves faster convergence. Th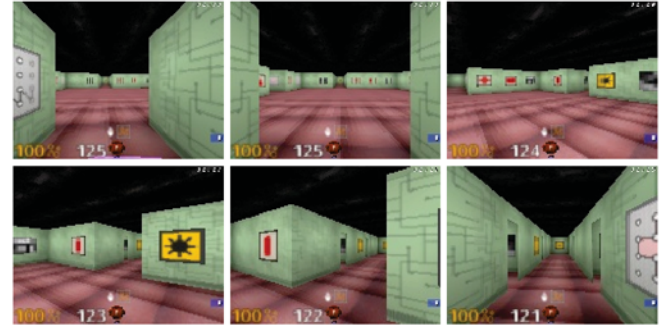erefore, we use BW images to train all behaviors, except **lmpd** that uses RGB images. We normalize each input image by subtracting the mean and dividing by the standard deviation, both estimated over the training set. We use Adam as the optimization tool, keeping the recommended parameter values [48]. We use randomly sampled batches of 256 instances. Initial learning rate is set to 10^{-4}. Also, we apply batch-normalization [49] after every convolutional layer (except the last one).

Table I provides details about the datasets used to learn each behavior. To collect this data, we use virtual environments to render expert trajectories consisting of the robot executing each behavior. As an example, Figure 7 shows a set of frames corresponding to a training instance used to fit the behavior *cross-hall*. In this case, the robot crosses the hall, following afterwards the right side corridor. As shown in Table I, the size of each dataset is variable and mostly depends on the expected training complexity of each behavior.

Navigational Behaviors				
	Train: #paths	Train: #images	Test: #trials	Accuracy
cf	150	46131	100	100%
io	8000	221734	100	100%
oo	8000	259937	100	67%
oo-b	8000	259937	100	96%
ch	3000	248950	100	100%
cc	4000	123653	100	100%
Perceptual Behaviors				
	-	Train: #images	Test: #images	Class. Acc.
pd	-	12945	1500 (Imgs)	98.2%
lmpd	-	38459	1500 (Imgs)	96.7%

TABLE I: Details of the datasets used to train each behavior as well as the accuracy achieved on an independent test set. For navigational behaviors, we measure accuracy as the number of trials where the virtual robot successfully complete the goal of the behavior, ex. robot successfully leaves an office using the intended departing direction. For perceptual behaviors, we measure accuracy as the successful classification rate.

In terms of navigational behaviors, Table I considers accuracy as the percentage of trials where the robot successfully completed the intended behavior, ex. crossing a corridor successfully using the intended departing direction. In terms of perceptual behaviors, accuracy is measured as the percentage of images where the model outputs a correct classification. For most behaviors, results in Table I indicate a

robust operation. The exception is behavior *out-of-office* (**oo**) that only succeed in 67% of the trials. This poor performance is mainly due to the random scheme used to initialize the robot position, which causes that the robot occasionally starts in a position directly facing a wall. In these cases, one will expect that the robot consistently rotates in one direction, searching for the position of the door to leave the office. However, the memoryless property of the CNN that is used to implement the behavior can not handle this capability. As a result, robot usually starts wandering close to the wall entering a deadlock situation. While it is possible to use a recurrent model to implement this behavior, we decide to restrict the initial position of the robot in such a way that it always starts facing the initial direction of the path of the expert, which always points towards the office door. Introducing this simplification, we obtain behavior **oo-b** that is able to achieve high robustness.

B. Behavior Integration

Next, we test the performance of the proposed approach to solve navigational tasks that require the integration of several behaviors. To conduct this experiment, we randomly generate 100 maps. For each map, we randomly select 10 navigational tasks consisting on moving between 2 different offices. This situation is similar to the case depicted in Figure 2-b, where the robot needs to navigate from office O_1 to office O_8. For simplicity, we directly extract the graph representation from the known configuration of the environment. In a more general case, this map can be inferred by a guided presentation of the environment to the robot or by an initial unsupervised exploration phase.

Table II shows our results. Accuracy is measured as the percentage of missions where the robot successfully reaches the intended goal location. As expected, the robot is able to successfully finish its mission in a significant proportion of the cases. Furthermore, the average length of 8.3 activations of navigational behaviors to complete each mission shows that the proposed approach is effective to control the selection and switching of behaviors. In terms of failure cases, a manual analysis reveals that these are mainly due to two problems. First, during robot navigation, the landmark detection behavior (**lmpd**) occasionally fails to detect the landmark that identifies the entrance to the goal office. As a result, the robot misses the activation of behavior *enter-office* (**io**), traversing its last corridor until its end, to finally wonder without destination. This problem is related to the use of a first person camera view that is always pointing to the current heading direction of the robot. As a result, occasionally, this camera is not pointing in a suitable direction missing the detection of a relevant landmark. A second failure case is due to the presence of repeated textural patterns in the virtual environment. In particular, the visual appearance of the view of a door just before leaving an office is highly similar to the view of the front door after leaving the office, as a result in some occasions the robot takes a wrong action, such as turning to face a hypothetical corridor before leaving the office. In practice, we believe that in a real implementation the previous two problems can be solved, and they do not represent a major limitation to the viability of our approach.

#Maps	#Missions/map	#Missions	Av. Steps/mission	Acc.
100	10	1000	8.3	81.2%

TABLE II: Performance of the proposed approach when robot needs to complete a mission consisting of navigating between 2 different offices. Accuracy is measured as the percentage of missions where the robot successfully reaches the intended goal location. We also report the average number of steps to complete a mission, which is equivalent to the average number of navigational behaviors needed to reach the goal location.

V. CONCLUSIONS

We present a new approach for indoor robot navigation that uses a graph representation to integrate perceptual and navigational behaviors. This approach offers an alternative view that departs from current methods and representations used to solve the robot navigation problem. As we discuss, current approaches focus on encoding geometry, as a way to facilitate robot localization and from there to select suitable actions to achieve navigational goals. In contrast, the proposed approach focuses on encoding suitable behaviors that directly guide the robot to complete its navigational goals. As a result, the proposed method offers two main advantages. On one hand, it leads to a robot navigational scheme that is less prompt to localization errors (ex. due to changes in the environment). On other hand, it relies on a semantically rich and compact representation of the world that facilitates the interaction with human users as well as the application of planning techniques.

An important aspect of the proposed approach is the determination of a set of behaviors that allow the robot to complete its navigational goals. In this work we show that, in simple scenarios, manual selection can lead to a working solution. Further research is needed to devise automatic methods that can deal with more complex situations. Recent work on modular visual reasoning [14] appears as an attractive option to fulfill this goal.

A relevant observation of the experimental validation is the limitation of pure reactive behaviors to successfully deal with the wide variety of situations faced by a purposeful robot. The inclusion of an extra orientation input helps us to discriminate between cases where a specific subgoal governs the overall execution of a specific behavior, such as leaving a hall using the left or right direction. However, the failure of the behavior *out-of-office* indicates the need to include models that can handle short-term memories.

The proposed approach opens several interesting research avenues. As an example, the current implementation separates the mapping and planning steps, we believe that it is possible to face these problems under a common learning framework. Recent work on the so-called Neural Turing Machine [47] appears as an attractive option to joint perception, representation, and control under a common framework. Also, the integration of the proposed representation with a natural language model is an interesting research avenue [50]. This can take advantage of the high-level semantic behind the proposed representation, leading to a more natural way to implement robot-human interaction.

Finally, it is important to notice that while our current implementation introduces certain simplifications to facilitate the operation of navigational and perceptual behaviors, we believe that the fundamental ideas behind the proposed

approach still hold in a more general case. This is in part supported by current success exhibited by deep learning techniques, which have demonstrated exceeding robustness to deal with complex perception and control tasks in natural environments. In this sense, a pending challenge is the implementation of the proposed approach using a real robot.

VI. APPENDIX: NOTATION

Code	Description
ool	out of office, take left
oor	out of office, take right
cf	follow-corridor
iol	enter office to the left
ior	enter office to the right
chs	cross hall, continue straight
chl	cross hall, take left
chr	cross hall, take right
ccc	change corridor, straight
ccl	change corridor to the left
ccr	change corridor to the right

TABLE III: Summary of the navigational behaviors that we use in our implementation.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. New York: Cambridge University Press, 2006.

[2] A. Araneda, S. Fienberg, and A. Soto, "A statistical approach to simultaneous mapping and localization for mobile robots," *Annals of Applied Statistics*, vol. 1, no. 1, pp. 66–84, 2007.

[3] A. Davison, I. Reid, N. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *PAMI*, vol. 29, no. 6, pp. 1052–1067, 2007.

[4] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. Rendon-Mancha, "Visual simultaneous localization and mapping: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, 2012.

[5] I. Kostavelis and A. Gasteratos, "Semantic mapping for mobile robotics tasks: A survey," *Robotics and Autonomous Systems*, vol. 66, no. 103, pp. 86–103, 2015.

[6] S. Bowman, N. Atanasov, K. Daniilidis, and G. Pappas, "Probabilistic data association for semantic SLAM," in *ICRA*, 2017.

[7] B. Kuipers, "Modeling spatial knowledge," *Cognitive Science*, vol. 2, pp. 129–153, 1978.

[8] J. Gibson, *The ecological approach to visual perception*. L. Erlbaum Associates, 1986.

[9] R. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal on Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986.

[10] I. Horswill, "Polly: A vision-based artificial agent," in *AAAI*, 1993.

[11] M. Bojarski, D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars." http://arxiv.org/abs/1604.07316, 2016.

[12] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *ICRA*, 2017.

[13] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Neural module networks," in *CVPR*, 2016.

[14] J. Johnson, B. Hariharan, L. van der Maaten, J. Hoffman, L. Fei-Fei, C. Zitnick, and R. Girshick, "Inferring and executing programs for visual reasoning," in *CVPR*, 2017.

[15] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[16] G. Walter, "An imitation of life," *Scientific American*, vol. 182, no. 5, 1950.

[17] R. Brooks, "A robot that walks; emergent behaviors from a carefully evolved network," *MIT AI Lab, Memo 1091*, 1989.

[18] R. Smith, M. Self, and P. Cheeseman, *Aut. Robot Vehicles*. Springer-Verlag, 1990, ch. Estimating Uncertain Spatial Relationships in Robotics, pp. 167–193.

[19] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *18th National Conference on Artificial Intelligence*, 2002.

[20] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Transactions on Intelligent Transportation Systems Magazine*, vol. 2, pp. 31–43, 2010.

[21] S. Thrun and M. Montemerlo, "The graph SLAM algorithm with applications to large-scale mapping of urban structures," *Int. J. Rob. Res.*, vol. 25, no. 5-6, pp. 403–429, 2006.

[22] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *Int. Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, 2006.

[23] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Fast incremental smoothing and mapping with efficient data association," in *ICRA*, 2007.

[24] E. Olson, "Real-time correlative scan matching," in *ICRA*, 2009.

[25] M. Cummins and P. Newman, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[26] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an RGB-D camera," in *Proceedings of the International Symposium of Robotics Research (ISRR)*, 2011.

[27] D. Gálvez-Lopez, M. Salas, J. Tardós, and J. Montiel, "Real-time monocular object SLAM," *Journal Robotics and Autonomous Systems*, vol. 75, pp. 435–449, 2016.

[28] P. Espinace, T. Kollar, N. Roy, and A. Soto, "Indoor scene recognition by a mobile robot through adaptive object detection," *Robotics and Autonomous Systems*, vol. 61, no. 9, 2013.

[29] I. Reid, "Towards semantic visual SLAM," in *Int. Conf. on Control Automation Robotics Vision (ICARCV)*, 2014.

[30] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, "Cognitive mapping and planning for visual navigation," in *CVPR*, 2017.

[31] E. Parisotto and R. Salakhutdinov, "Neural map: Structured memory for deep reinforcement learning." http://arxiv.org/abs/1702.08360, 2017.

[32] A. Elfes, "Occupancy grids: A probabilistic framework for robot perception and navigation," Ph.D. dissertation, Department of Electrical and Computer Engineering, Carnegie Mellon University, 1989.

[33] J. Oh, V. Chockalingam, S. Singh, and H. Lee, "Control of memory, active perception, and action in minecraft," in *ICML*, 2016.

[34] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in *NIPS*, 2015.

[35] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, D. Kumaran, and R. Hadsell, "Learning to navigate in complex environments," in *ICLR*, 2017.

[36] J. O'Keefe and L. Nadel, *The Hippocampus as a Cognitive Map*. Oxford University Press, UK, 1978.

[37] T. Hafting, M. Fyhn, S. Molden, M. Moser, and E. Moser, "Microstructure of a spatial map in the entorhinal cortex," *Nature*, vol. 36, no. 11, pp. 801–806, 2005.

[38] D. Pomerleau, *Neural Network Perception for Mobile Robot Guidance*. Boston: Kluwer Academic Publishers, 1993.

[39] P. Sermanet, K. Xu, and S. Levine, "Unsupervised perceptual rewards for imitation learning," in *Robotics: Science and Systems*, 2017.

[40] A. Giusti, J. Guzzi, D. Ciresan, F. He, J. Rodriguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. Di-Caro, D. Scaramuzza, and L. Gambardella, "A machine learning approach to visual perception of forest trails for mobile robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, 2016.

[41] I. Armeni, A. Sax, A. Zamir, and S. Savarese, "Joint 2d-3d-semantic data for indoor scene understanding," in *CVPR*, 2017.

[42] C. Beattie, J. Leibo, D. Teplyashin, T. Ward, M. Wainwright, H. Küttler, A. Lefrancq, S. Green, V. Valdés, A. Sadik, J. Schrittwieser, K. Anderson, S. York, M. Cant, A. Cain, A. Bolton, S. Gaffney, H. King, D. Hassabis, S. Legg, and S. Petersen, "Deepmind lab," in *http://arxiv.org/abs/1612.03801*, 2016.

[43] J. Latombe, *Robot motion planning*. Kluwer Academic, 1991.

[44] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. of Robotics Research*, vol. 30, no. 7, 2011.

[45] I. Şucan, M. Moll, and L. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[46] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICRL*, 2015.

[47] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines." http://arxiv.org/abs/1410.5401, 2014.

[48] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *https://arxiv.org/abs/1412.6980*, 2014.

[49] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015.

[50] X. Zang, M. Vzquez, J. Niebles, A. Soto, and S. Savarese, "Behavioral indoor navigation with natural language directions," in *HRI*, 2018.