IQA: Visual Question Answering in Interactive Environments

Daniel Gordon[2] Aniruddha Kembhavi[1] Mohammad Rastegari[1]
Joseph Redmon[2] Dieter Fox[2] Ali Farhadi[1,2]
[1]Allen Institute for Artificial Intelligence
[2]Paul G. Allen School of Computer Science, University of Washington

Abstract

*We introduce Interactive Question Answering (*IQA*), the task of answering questions that require an autonomous agent to interact with a dynamic visual environment.* IQA *presents the agent with a scene and a question, like: "Are there any apples in the fridge?" The agent must navigate around the scene, acquire visual understanding of scene elements, interact with objects (e.g. open refrigerators) and plan for a series of actions conditioned on the question. Popular reinforcement learning approaches with a single controller perform poorly on* IQA *owing to the large and diverse state space. We propose the Hierarchical Interactive Memory Network (*HIMN*), consisting of a factorized set of controllers, allowing the system to operate at multiple levels of temporal abstraction, reducing the diversity of the action space available to each controller and enabling an easier training paradigm. We introduce* IQADATA*, a new Interactive Question Answering dataset built upon AI2-THOR, a simulated photo-realistic environment of configurable indoor scenes [95] with interactive objects.* IQADATA *has 75,000 questions, each paired with a unique scene configuration. Our experiments show that our proposed model outperforms popular single controller based methods on* IQADATA*. For sample questions and results, please view our video: https://youtu.be/pXd3C-1jr98.*

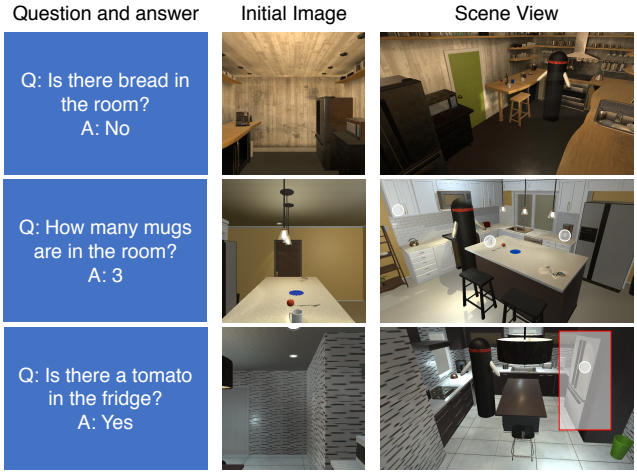

Figure 1. This figure illustrates a few samples in the Interactive Question Answering Dataset. Each row shows a question paired with the initial view of the agent and a scene view of the environment. The scene view is only displayed for illustration purposes and is not available to the agent. The agent is shown in black, with the red line indicating the agent's camera direction. The locations of the objects of interest for each question are outlined. Note that none of the questions can be answered given only the initial view.

1. Introduction

A longstanding goal of the artificial intelligence community has been to place agents in the real world that can perform tasks to aid humans and communicate with them via natural language. For instance, a household robot might be posed the following questions: *Do we need to buy more milk?* which would require it to navigate to the kitchen, open the fridge and check to see if there is sufficient milk in the milk jug, or *How many packets of cookies do we have?* which would require the agent to navigate to the cabinets, open several of them and count the number of cookie packets. Towards this goal, Visual Question Answering (VQA), the problem of answering questions about visual content, has received significant of attention from the computer vision and natural language processing communities. While there has been a lot of progress on VQA, research by and large focuses on answering questions passively about visual content, i.e. without the ability to interact with the environment generating the content. An agent that is only able to answer questions passively is clearly limited in its capacity to aid humans in their tasks.

We introduce Interactive Question Answering (IQA), the task of answering questions that require the agent to interact with a dynamic environment. IQA poses several key challenges in addition to the ones posed by VQA. **First**, the agent must be able to navigate through the environment. **Second**, it must acquire an understanding of its environ-

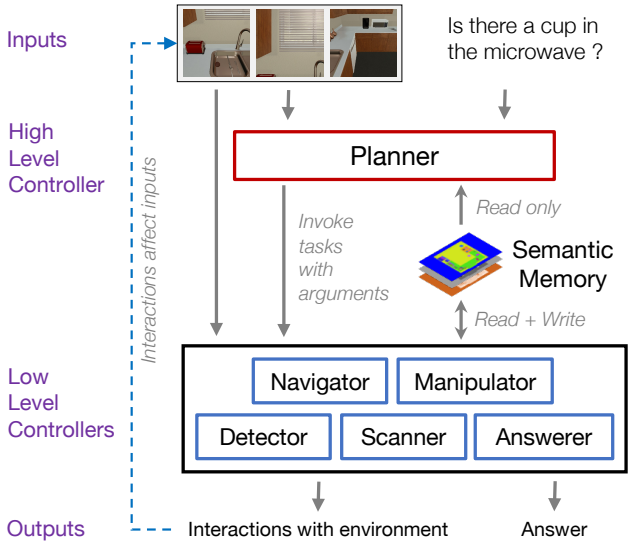

Figure 2. An overview of the Hierarchical Interactive Memory Network (HIMN)

ment including objects, actions and affordances. **Third**, the agent must be able to interact with objects in the environment (such as opening the microwave, picking up books, etc.). **Fourth**, the agent must be able to plan and execute a series of actions in the environment conditioned on the questions asked off it. Past approaches to similar problems (such as visual semantic planning [93] and task oriented language grounding [8, 21]) have used reinforcement learning to train a single agent to perform tasks. While this paradigm with a single controller works well when the action space is limited and sequences are short, our experiments show that it does not scale well for IQA, where the cardinality and diversity of the the set of possible actions is much larger (navigation, interaction and unique answer choices), resulting in poor QA accuracies as well as poor generalization to unseen environments.

To address these challenges, we propose HIMN (Hierarchical Interactive Memory Network). Figure 2 provides an overview of HIMN. Akin to past works on hierarchical reinforcement learning, HIMN is factorized into a hierarchy of controllers, allowing the system to operate, learn, and reason across multiple time scales while simultaneously reducing the complexity of the tasks responsible to each controller. A high level controller, referred to as the *Planner* chooses the task to be performed (for example, navigation / manipulation / answering / etc.) and generates attributes for the chosen task. Tasks specified by the *Planner* are executed by a set of low level controllers (*Navigator*, *Manipulator*, *Detector*, *Scanner* and *Answerer*) which return control to the *Planner* when a task termination state is reached. For instance, the *Planner* might choose the task of navigation with attributes $(4, 1)$ indicating the coordinates of a target location relative to the current state of the agent. This invokes the *Navigator* which attempts to move the agent towards this target. The *Navigator* in turn invokes other low level controllers such as the *Scanner* and *Detector* which enables it to gather data about the scene and avoid obstacles. The *Navigator* terminates the task either when it gets to the target or a nearby location if it determines that it is unable to complete the task successfully. Factorizing the model into a hierarchy of controllers partitions the action space as well as simplifies the tasks that any one controller is expected to perform. This allows us to pre-train each controller independently and efficiently, while assuming oracle versions of the remaining controllers. Our experiments show that this factorization enables higher accuracy and generalization to unseen scenes.

Several question types require the agent to *remember* where it has been and what it has seen. For example, *How many pillows are in this house?* requires an agent to navigate around the rooms, open closets and keep track of the number of pillows it encounters. For sufficiently complex spaces, the agent needs to hold this information in memory for a long time. This motivates the need for an explicit external memory representation that is filled by the agent as it interacts with its environment. This memory must be both spatial and semantic so it can represent *what* is *where*. We propose a new recurrent layer formulation: Egocentric Spatial GRU (esGRU) to represent this memory (Sec 4.1). The esGRU only allows writing to a local spatial window within the memory, dependent on the agent's current location and viewpoint. However, the entire memory is available while planning tasks and answering questions. This speeds up computations and prevents corrupting the memory at locations far away from the agent's current viewpoint.

Training and evaluating interactive agents in the real world is currently prohibitive from the standpoint of operating costs, scale and research reproducibility. A far more viable alternative is to train and evaluate such agents in realistic simulated environments. Towards this end, we present the Interactive Question Answering Dataset (IQADATA) built upon AI2-THOR [95], a photo-realistic customizable simulation environment for indoor scenes integrated with the Unity [1] physics engine. IQADATA consists of over 75,000 multiple choice questions, each question accompanied by a unique scene configuration. Question types include Existence, Counting, and Spatial Relationships between objects. IQADATA is a balanced dataset that prevents models from obtaining high accuracies by simply exploiting trivial language and configuration biases. IQADATA will be released publicly.

We evaluate HIMN on IQADATA using a question answering accuracy metric and show that it outperforms a baseline based on a common architecture for reinforcement learning used in past works. We also analyze the frequency of making invalid actions to better understand the choices

made before answering.

In summary, our contributions include: (a) proposing Interactive Question Answering, the task of answering questions that require the agent to interact with a dynamic environment, (b) presenting the Hierarchical Interactive Memory Network, a question answering model factorized into a high level *Planner*, a set of low level controllers and a rich semantic spatial memory, (c) the Egocentric Spatial GRU, a new recurrent layer to represent this memory and (d) a new dataset IQADATA towards the task of IQA.

2. Related Work

Visual Question Answering (VQA): VQA has seen a lot of progress over the past few years, owing to the design of deep architectures suited for this task and the creation of large VQA datasets to train these models [87]. These include datasets of natural images [2, 16, 40, 49, 84, 94], synthetic images [2, 3, 26, 30, 31], natural videos [25, 79], synthetic videos [36, 58] and multimodal contexts [32]. Some of these use questions written by humans [2, 31, 32, 40, 94] and others use questions that are generated automatically [3, 26, 30]. IQADATA is set in a photo-realistic simulation environment and uses automatically generated questions. In contrast to the aforementioned QA datasets that only require the agent to observe the content passively, IQADATA requires the agent to interact with a dynamic environment.

The first deep architectures designed for VQA involved using an RNN to encode the question, using a CNN to encode the image and combining them using fully connected layers to yield the answer [2, 50]. Attention models have proven to be very popular for VQA with several variants employed [15, 48, 90, 91, 92, 94]. More recently, modular networks [3, 22, 27] that construct an explicit representation of the reasoning process by exploiting the compositional nature of language have been proposed. Similar architectures have also been applied to the video domain with extensions such as spatiotemporal attention [25, 58]. Other bodies of research involve the use of external knowledge bases [83, 84, 88] and explicit scene parsing [82] to answer visual questions. Our proposed approach to question answering allows the agent to interact with its environment and is thus fundamentally different to past QA approaches. However, we note that approaches such as visual attention and modularity can easily be combined with our model to provide further improvements.

Reinforcement Learning (RL): RL algorithms have been employed in a wide range of problems including locomotion [37], learning motor primitives [45, 64], obstacle detection [53] and autonomous flight [38, 70]. Recent works employing deep architectures train large networks that can learn policies to directly map pixels to actions [46, 54, 55, 73, 74]. For more comprehensive surveys of pre-deep and post-deep era applications of RL techniques, we refer the reader to [4, 28].

Of particular relevance to our proposed method to QA is the area of hierarchical reinforcement learning (HRL), which consists of a high level controller and one or more low level controllers. The high-level controller selects a subtask (or a subtask-parameter pair) to be executed, and invokes one of the low level controllers. The advantage of hierarchical RL is that it allows the model to operate at multiple levels of temporal abstraction. Early works proposing hierarchical RL algorithms include [11, 63, 78]. More recent approaches in the deep learning era include [41] who propose hierarchical-DQN with an intrinsically motivated RL algorithm, [80] who use HRL to create a lifelong learning system that has the ability to reuse and transfer knowledge from one task to another, and [60] who use HRL to enable zero shot task generalization by learning subtask embeddings that capture correspondences between similar subtasks. Our use of HRL primarily lets us learn at multiple time scales and its integration with the semantic memory lets us divide the complex task of IQA into more concrete tasks of navigation, detection, planning etc. that are easier to train.

RL techniques have recently been applied to QA tasks, most notably by [27] to train a program generator that constructs an explicit representation of the reasoning process to be performed and an execution engine that executes the program to predict the answer.

Visual Navigation: Navigation is a core ingredient in our approach. In order to interact with the objects in the virtual environment, an agent must navigate through the environment. There is a large body of work on visual navigation. The majority of them fall into two categories: offline map-based and online map-based. Offline map-based techniques [6, 7, 35, 61] require the complete map of the environment to make any decisions about their actions, which limits their use in unseen environments. Online map-based methods [10, 13, 59, 75, 81, 85] often construct the map while exploring the environment. The majority of these approaches use the computed map for navigation only, whereas our model constructs a rich semantic map which is used for navigation as well as planning and question answering. Other navigation methods include map building guided by humans [34, 69] and map-less approaches [19, 24, 39, 43, 47, 52, 66, 71, 95] which use techniques such as obstacle avoidance and feature matching and depend upon implicit representations of the world to perform navigation. Recently Gupta *et al*. [18] proposed a joint architecture for a *mapper* that produces a spatial memory and a *planner* that can plan paths. The similarities between our works lie in the usage of a hierarchical system and a spatial memory. In contrast to their work, navigation is not the end goal of our system, but a subtask towards the end goal of question answering and our action space is more diverse

as it includes interaction and question answering. Finally, our use of a simulated environment allows us to produce a large set of novel scene configurations and have access to automatic ground truth annotations, which can be used for training and ablation studies (Sec 5).

Visual Planning: To answer questions such as *Do I need to buy milk?* an agent needs to plan a sequence of actions to explore and interact with the environment. A large body of research on planning algorithms [12, 14, 29, 76, 77] work with high-level formal languages. These techniques are designed to handle low-dimensional state spaces but do not scale well in the presence of high dimensional state spaces such as ours.

The most relevant works to IQA are the problems of visual navigation by [95] and visual semantic planning by [93] in the AI2-THOR environment. The former only tackles navigation, and the latter only focuses on high level planning and assumes an ideal low level task executor; in contrast, our model trains low level and high level controllers jointly. Also, both these approaches do not generalize well to unseen scenes, whereas our experiments show that we do not overfit to previously encountered environments. Finally, these methods do not build an explicit semantic map whereas we do, which helps us navigate as well as answer questions.

Recently Chaplot *et al*. [8] and Hill *et al*. [21] have proposed models to complete navigation tasks specified via language (e.g. *Go to the red keycard*) and trained their systems in simulated 3D environments. These models show the ability to generalize to unseen instructions of seen concepts. In contrast, we tackle several question types, that require a variety of navigation behaviours and interaction, our model is hierarchical, we use an explicit rich semantic memory and the environment we use is significantly more photo-realisitic. Finally, we compare our proposed HIMN model to a baseline (A3C in Section 5) system that very closely resembles the model architectures proposed in [8] and [21].

Visual Learning by Simulation: There has been an increased use of simulated environments and game platforms to train computer vision systems to perform tasks such as learning the dynamics of the world [56, 57, 86], semantic segmentation [20], pedestrian detection [51], pose estimation [62] and urban driving [9, 67, 68, 72]. Several of these are also interactive making them suitable to learn control, including [5, 33, 44, 89, 95]. We choose to use AI2-THOR [95] in our work since it provides a photo-realistic and interactive environment of real world scenes, making it very suitable to train QA systems that might be transferable to the real world.

Interactive Question Answering Dataset Statistics		
	Train	Test
Existence	25,200	560
Counting	25,200	560
Spatial Relationships	25,200	560
Rooms	25	5
Total scene configurations (s.c.)	75,600	1,680
Avg # objects per (s.c.)	46	41
Avg # interactable objects (s.c.)	21	16

Table 1. This table shows the statistics of our proposed dataset in a variety of question types, objects and scene configurations.

3. Learning Framework

3.1. Actionable Environment

Training and evaluating interactive agents in the real world is currently prohibitive from the standpoint of operating costs, scale, time and research reproducibility. A far more viable alternative is to use simulated environments. However, the framework should be visually realistic, allow interactions with objects, and have a detailed model of the physics of the scene so that agent movements and object interactions are properly represented. Hence, we adopt the AI2-THOR environment [95] for our purposes. AI2-THOR is a photo-realistic simulation environment of 120 rooms in indoor settings, tightly integrated with a physics engine. Each scene consists of a variety of objects, from furniture such as couches, appliances such as microwaves and smaller objects such as crockery, cutlery, books, fruit, etc. Many of these objects are actionable such as fridges which can be opened, cups which can be picked up and put down, and stoves which can be turned on and off.

3.2. Interactive Question Answering Dataset

IQADATA is a question answering dataset built upon AI2-THOR. It consists of over 75,000 multiple choice questions. Table 1 shows statistics for IQADATA. Each question is accompanied with a unique scene configuration drawn from one of 30 kitchen rooms in AI2-THOR, with a unique arrangement of objects in it. The wide variety of configurations in IQADATA prevent models from memorizing simple rules like "apples are always in the fridge" and render this dataset challenging. IQADATA consists of several question types including: Existence questions (*Is there an apple in the kitchen?*), Counting questions (*How many forks are present in the scene?*), and Spatial Relationship questions (*Is there lettuce in the fridge? / Is there a cup on the counter-top?*). Questions and answer choices are generated automatically. Since natural language understanding is not a focus of this dataset, questions are generated using a set of templates written down apriori. Extending IQADATA to include more diverse questions generated by turkers is fu-

ture work. IQADATA is a balanced dataset that prevents models from obtaining high accuracies by simply exploiting trivial language and scene configuration biases. Similar to past balanced VQA datasets [17], each question is associated with multiple scene configurations that result in different answers to the question. We split the 30 kitchen rooms into 25 train and 5 test, and have 1008 unique "question-scene configuration" pairs for each room-question type pair in train, and 112 in test. An episode is finished when the *Answerer* is invoked. We evaluate different methods using Top-1 accuracy. Figure 1 shows example questions with scenes and starting viewpoints of the agent.

3.3. Agent and Objects

The agent in our environments has a single RGB camera mounted at a fixed height. An agent can perform one of five navigation actions (move ahead 25 cm, rotate 90 degrees left or right, look up or down 30 degrees). We assume a grid-world floor plan that ensures that the agent always moves along the edges of a grid and comes to a stop on a node in this grid. The agent can perform two interaction actions (open and close) to manipulate objects. A wide variety of objects (fridges, cabinets, drawers, microwaves, etc.) can be interacted with. If there are multiple items in the current viewpoint which can be opened or closed, the environment chooses the one with the largest area in pixels within the viewpoint. The success of each action depends on the current state of the environment as well as the agent's current location. For instance, the agent cannot open a cabinet that is more than 1 meter away or is not in view, or is already open, and it cannot walk through a table or a wall.

4. Model

We propose HIMN (Hierarchical Interactive Memory Network), consisting of a hierarchy of controllers that operate at multiple levels of temporal abstraction and a rich semantic memory that aids in navigation, interaction, and question answering. Figure 2 provides an overview of HIMN. A high level controller, referred to as the *Planner* chooses the task to be performed (e.g. navigation, manipulation, answering) and generates arguments for the chosen task (e.g. open this cabinet). Tasks specified by the *Planner* are executed by a set of low level controllers (*Navigator*, *Manipulator*, *Detector*, *Scanner* and *Answerer*) which return control to the *Planner* when a task termination state is reached. We now describe each of these components in greater detail.

4.1. Spatial Memory

Several question types require the agent to keep track of objects that it has seen in the past along with their locations. For complex scenes with several locations and interactable objects, the agent needs to hold this information

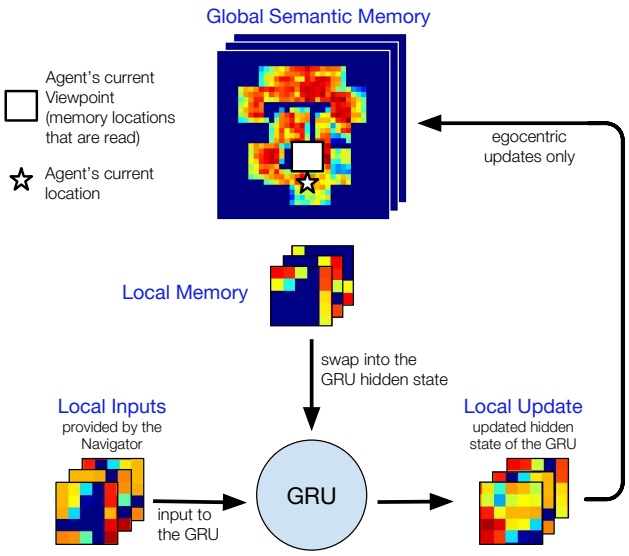

Figure 3. An overview of the Egocentric Spatial GRU (esGRU): The esGRU only allows writing to a local window within the memory, dependent on the agent's current location and viewpoint.

in memory for a long duration. This motivates the need for an explicit external memory representation that is filled by the agent on the fly and can be accessed at any time. To address this, HIMN uses a rich semantic spatial memory that encodes a semantic representation of each location in the scene. Each location in this memory consists of a feature vector encoding object detection probabilities, free space probability (a 2D occupancy grid), coverage (has the agent inspected this location before), navigation intent (has the agent attempted to visit this location before) as well as the agent's current pose. We propose a new recurrent layer formulation: Egocentric Spatial GRU (esGRU) to represent this memory, illustrated in Figure 3. The esGRU maintains an external global spatial memory represented as a 3D tensor. At each time step, the esGRU swaps in local egocentric copies of this memory into the hidden state of the GRU, performs computations using current inputs, and then swaps out the resulting hidden state into the global memory at the predetermined location. This speeds up computations and prevents corrupting the memory at locations far away from the agent's current viewpoint. Yet when planning for a task or answering questions, the agent can access the full memory, enabling long-term recall from observations seen hundreds of states prior. Furthermore, only low level controllers have read-write access to this memory. Since the *Planner* only makes high level decisions, without interacting with the world at a lower level, it only has read access to the memory.

4.2. Planner

The high level *Planner* invokes low level controllers in order to explore the environment, gather knowledge needed

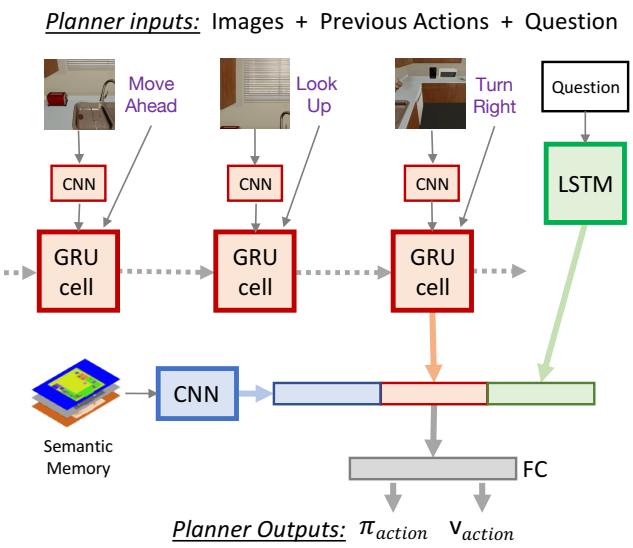

Figure 4. Schematic representation of the *Planner*

to answer the given question, and answer the question. We frame this as a reinforcement learning problem where the agent must issue the fewest possible commands that result in a correct answer. The agent must learn to explore relevant areas of the scene based on learned knowledge (e.g. apples are often in the fridge, cabinets are openable, etc.), the current memory state (e.g. the fridge is to the left), current observations (e.g. the fridge is closed) and the question. At each timestep, the agent produces a policy π consisting of probabilities π_i for each action, and a value v for the current state. π and v are learned using the A3C algorithm [54]. Figure 4 shows a schematic of the *Planner*. It consists of a GRU which accepts at each time step the current viewpoint (encoded by a CNN) and the previous action. The *Planner* has read access to the semantic memory. The output of this GRU is combined with the question embedding and an embedding of the entire memory to predict π and v. The agent receives a fixed reward/penalty based answering correctly/incorrectly. It is also provided a constant time penalty to encourage efficient explorations of the environment and quick answering, as well as a penalty for attempting to perform invalid actions (see section 5.1 for more details). The agent is also given intermediate rewards for increasing the "coverage" of the environment, effectively training the network to maximize the amount of the room it has explored as quickly as possible. Finally, at each time step, the *Planner* also predicts which high level actions are viable given the current world state. In many locations in the scenes, certain navigation destinations are unreachable or there are no objects to interact with. Predicting possible/impossible actions at each time step, allows gradients to propagate through all actions rather than just the chosen action. This leads to higher accuracies and faster convergence.

4.3. Low level controllers

Navigator The *Navigator* is invoked by the *Planner* which also provides it with the relative coordinates of the target location. The *Navigator* reads and writes data into the free space map of the memory. Given a destination specified by the *Planner* and the current occupancy grid, we run A* search to find the shortest path to the goal. As the *Navigator* moves through the environment, it updates the occupancy estimates based on current observations using the esGRU and recomputes the shortest path. It also invokes the *Scanner* controller to obtain a wide angle view of the environment. Given that the requested destination may be outside the bounds of the room or otherwise impossible (e.g. at a wall or other obstacle), the *Navigator* also predicts a termination signal, and then returns control to the *Planner*.

Scanner The *Scanner* is a simple controller which captures images by rotating the camera up and down while maintaining the agent's current location. The *Scanner* calls the *Detector* on every new image.

Detector Object detection is a critical component of HIMN, given that all questions in IQADATA involve one or more objects in the room. We use YOLO V2 [65] fine-tuned on the AI2-THOR training scenes as an object detector. We estimate the depth of an object using the FRCN depth estimation network [42] and project the probabilities of the detected objects onto the ground plane. Both of these networks operate at real-time speeds, which is necessary since they are invoked at every timestep. The detection probabilities are incorporated into the spatial memory using a moving average update rule. We also perform experiments where we substitute the trained detector and depth estimator with oracle detections provided by the environment. However, the oracle is limited to a 1 meter semicircle in front of the agent and cannot see through objects such as closed doors or walls. Furthermore, detections provided by the environment still requires the network to learn affordances. For instance, the network must learn that *microwaves can be opened*, *apples fit in fridges*, etc.

Manipulator The *Manipulator* is invoked by the *Planner* to manipulate the current state of an object. For example, opening and closing the microwave. This leads to a change in the visual appearance of the scene. If the object is too far away or out of view, the action will fail and control will be returned to the *Planner*.

Answerer The *Answerer* is invoked by the *Planner* to answer the question. It uses the full spatial memory as well as the question embedding vector to predict an answer a to the question. The question vector is tiled to create a tensor with the same width and height as the spatial memory. These are depthwise concatenated with the spatial memory and passed through 4 convolution and max pool layers followed by a sum over the spatial layers. This output vector is fed through two fully connected layers and a softmax over

	Existence		Counting		Spatial Relationships	
Model	Accuracy	Length	Accuracy	Length	Accuracy	Length
Most Likely Answer Per Q-type (MLA)	57	-	27	-	58	-
A3C with no detections	56.9	5.26	26.42	2.56	**59.1**	5.01
A3C with YOLO [65] detections	54.29	8.75	26.78	12.13	44.64	9.4
HIMN with YOLO [65] detections	**63.39**	150.32	**35.89**	236.31	57.15	80.12
A3C with ground truth (GT) detections	59.51	5.93	27.13	2.51	**66.2**	5.69
HIMN with GT detection	**69.78**	48.25	**32.07**	104.86	65.54	90.26
HIMN with GT detection and oracle navigator (HIMN-GT)	73.03	38.25	45.35	50.83	71.42	37.31
HIMN-GT Question not given to planner	45.89	40.35	38.21	37.93	64.28	28.23
HIMN-GT No loss on invalid actions	62.67	8.88	20.71	7.18	62.32	7.71

Table 2. This tables compares the test accuracy and episode lengths of question answering across different models and question types.

possible answer choices. After the *Answerer* is invoked, the episode ends, whether the answer was correct or not.

4.4. Training

Our hierarchical structure allows us to train each low-level controller independently.

Navigator: The *Navigator* is trained by providing pairs of random starting points and random goal locations. At the beginning the agent assumes an empty room and follows the shortest path. At each step it tries to predict the occupancy map in a 5x5 grid in front of it using a CNN. The occupancy supervision is obtained from the environment. We train the *Navigator* using the sigmoid cross-entropy loss on each of the 5x5 grid locations. The logits are transformed into weights for the navigation graph using the formula $weight = max(1, 5e^x)$ ensuring each step costs at least 1.

Answerer: To train the *Answerer*, we need to mimic the spatial memory map that the agent would create while exploring the environment. Therefore, we create several random partially completed memory maps paired with questions and train the *Answerer* with a supervised loss.

Detector: The *Detector* is trained by fine-tuning YOLO-V2 [65] on the AI2-THOR training scenes. It is trained to identify small object instances which may appear in multiple scenes (apples, forks, etc.) as well as large object instances which only appear in a single scene (e.g. each fridge model will only exist in one scene).

Scanner and Manipulator: There are no trainable parameters for these controllers in our current setup. Their behavior is predefined by the AI2-THOR environment.

Planner: To train the *Planner*, we assume a perfect *Navigator* and *Detector* by using the ground truth shortest path for the *Navigator* and the ground truth object information for the *Detector*. To train the policy in the *Planner*, we use the A3C algorithm, providing a reward for answering correctly as well as a small reward for exploring new locations which encourages the agent to thoroughly explore the environment. In a training iteration in the A3C algorithm, the parameters will receive gradients only for the chosen action which slows down the convergence of the training. To facilitate the training of the *Planner*, at each step, we predict the viability of all of the high level actions, and train a supervised classifier. This allows us to propagate loss signal through all actions rather than only the chosen one. This supervision is also provided by the environment.

Training end-to-end: After training each component separately, we fine tune the *Planner* and the *Answerer* jointly, using inputs from the remaining controllers.

5. Experiments

We evaluate HIMN on the IQADATA dataset, using Top-1 question answering accuracy. An initial baseline of Most likely answer per Question-Type (MLA) shows any bias in the generated questions. The learned baseline (referred to as A3C) that we compare to, is based on a common architecture for reinforcement learning used in past works including for visual semantic planning [93] and task oriented language grounding [8, 21]. We extend this for the purpose of question answering. Since HIMN has access to object detections provided by either the environment or YOLO [65], we also provide these detections to the baseline. For the baseline model, at each time-step, the raw RGB image observed by the agent is concatenated depth wise with object detections projected onto the image plane. This tensor is passed through convolutional layers and fed into a GRU. The question is passed through an LSTM. The output of the LSTM and GRU are concatenated, and passed through two fully connected layers to produce probabilities π_i for each action and a value v. The output of the first fully connected layer is also passed to an answering module that consists of two more fully connected layers with a softmax on the space of all possible answers. The model is trained using A3C and a supervised loss on the answers. We also report results for A3C without using any object detector.

Table 2 shows the test accuracies and the average episode lengths for the proposed HIMN model and baselines for each question type. The MLA baseline shows that the dataset is

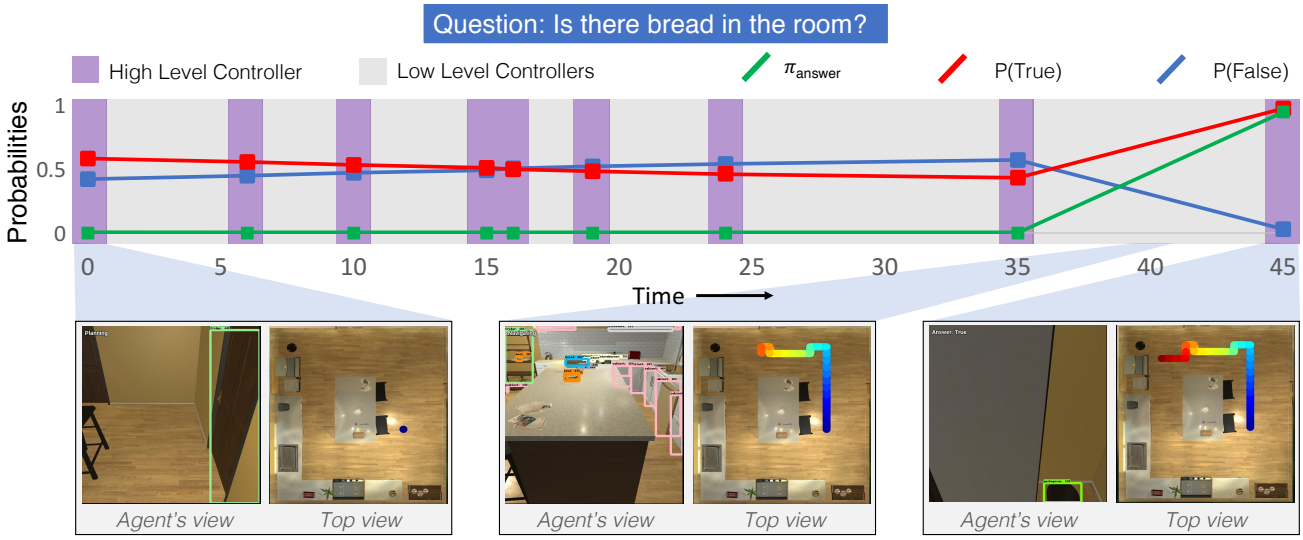

Figure 5. A sample trajectory for answering the existence question: *Is there bread in the room?* The purple sections indicate a *Planner* step, and the gray sections indicate a lower level controller such as the *Navigator* is controlling the agent. For a more detailed explanation, refer to section 5.3.

fairly balanced overall. HIMN significantly outperforms the A3C baselines on two of the three question types, both with YOLO object detections as well as ground truth object detections. For existence and counting, A3C is close to the MLA baseline. The A3C methods also tend to take very few steps before answering the question, so few that questions are often impossible to answer with any degree of certainty. HIMN, on the other hand, takes significantly longer to answer, but spends its time exploring the room so that it can answer more accurately. This indicates that HIMN (which uses an explicit semantic spatial memory with egocentric updates) is more effective than A3C (which uses a GRU) at (a) Estimating when the environment has been sufficiently explored, given the question (b) Keeping track of past observations for much longer durations, which is important in determining answers for questions that require a thorough search of the environment (c) Keeping track of multiple object instances in the scene, which may be observed several time steps apart (which is crucial for answering counting questions).

For spatial relationship questions, HIMN does not outperform the baseline methods. However the baselines are not significantly more effective than guessing the most probable answer, indicating that all methods struggle with this question type. With ground truth detections, both methods improve, indicating that accurate detections are especially useful for this question type. The training data for YOLO object detection contains objects that are far away as well as close to the agent. When a large object such as a fridge is very close to the agent, typically only a portion is within the camera's viewpoint and appears as a single solid surface. This leads to a significant number of false positives for containers such as fridges and cabinets, when the agent sees planar surfaces. Thus questions like *Is there an apple in the fridge?* may be mistakenly answered "yes" if a fridge is detected in the vicinity of an apple.

We further perform three ablative experiments on our network structure and inputs. First, we substitute our learned navigation controller with an oracle *Navigator* using the shortest viable path in the environment. When the optimal *Navigator* is provided, HIMN further improves over the baseline. This is because the *Planner* can more accurately direct the agent through the environment, allowing it to be both more efficient and more thorough at exploring the environment. It also takes fewer invalid actions (as seen in table 3), indicating that it is less likely to get stuck in parts of the room. In our second ablative experiment, we remove the question vector from the input of the *Planner*, only providing it to the *Answerer*, which results in a noticeable drop in the accuracy. This shows that the *Planner* utilizes the question to direct the agent towards different parts of the room to gather information required to answer the question. For instance any question about an object in the fridge requires the planner to know the fridge needs to be opened. If the planner is not told the question, it has no reason to open the fridge, and instead will likely choose to continue exploring the room as exploration often gives more reward than opening an object. Also, some questions can be answered soon after an object is observed (*e.g.* Existence), whereas others require longer explorations (*e.g.* Counting). Having access to the questions can clearly help the *Planner* in these scenarios. Table 2 shows that HIMN does in fact explore the environment for longer durations for Counting questions as opposed to Existence questions, with Spatial Relationship

Percentage of invalid actions			
Model	Existence	Counting	Spatial Relationships
A3C with no detections	25.9	18.72	24.53
A3C with YOLO [65] detections	22.63	33.8	30.91
A3C with GT detections	28.86	22.59	27.22
HIMN No loss on invalid actions	22.66	23.28	22.26
HIMN with YOLO [65] detections	8.08	9.85	7.53
HIMN with GT detections	14.25	11.95	12.72
HIMN with Oracle Navigator	2.77	4.04	4.14

Table 3. This tables compares the percentage of invalid actions across different models on test. Lower is better.

questions lying in between the two. In our third experiment, we remove the loss on invalid actions. The results are shown in the last row of table 2. If we do not apply any loss on these actions and only propagate gradients through the chosen action, the agent suffers from the difficulty of exploring a large action space and acts much more like the A3C baseline. The validity loss also serves as an auxiliary task which has been shown to aid the convergence of RL algorithms [23].

5.1. Invalid Actions

Table 3 shows the percentage of invalid actions taken by the different methods. Failed actions are due to navigation failures (failing to see an obstacle) or interaction failures (trying to interact with something too far away or otherwise impossible). There is a clear benefit to including a loss on the invalid actions both in terms of QA accuracy, as can be seen in table 2, as well as in terms of percentage of valid actions performed, shown in table 3. All models in table 3 are penalized for every invalid action they attempt, but this only provides feedback on a single action at every timestep. With the addition of a supervised loss on all possible actions, the percentage of invalid actions performed is roughly halved. By directly training our agent to recognize affordances (valid actions), we are able to mitigate the difficulties posed by a large action space, allowing the *Planner* to learn much more quickly. By replacing the learned *Navigator* with an oracle, we observe that the majority of failed actions are due to navigation failures. We believe that with a smaller step size, we would further reduce the navigation errors at the expense of longer trajectories.

	Existence		Counting		Spatial Relationships	
	S	U	S	U	S	U
HIMN with YOLO [65] detections	68.12	63.39	35.33	35.89	55.56	57.15
HIMN with GT detections	73.91	69.78	25	32.07	74.49	65.54

Table 4. This tables compares the accuracy of question answering across different models on Seen (S) and Unseen (U) environments.

5.2. Generalization in Unseen Environments

One benefit of HIMN over other RL architectures is that encoding semantic information into a spatial map should generalize well in both seen and unseen environments. Despite our relatively small number of training rooms, table 4 shows that our method only loses up to a few percentage points of accuracy when tested on unseen environments. This contrasts with many other end-to-end RL methods which learn deep features that tend to limit their applicability outside a known domain [93, 95].

5.3. Qualitative Results

Figure 5 shows a sample run of HIMN for the question "Is there bread in the room." Initially, $P(True)$ and $P(False)$ both start near 50%. The *Planner* begins searching the room by navigating around the kitchen table. During the initial exploration phase, bread is not detected, and $P(False)$ slowly increases. At timestep 39, the *Navigator* invokes the *Detector*, which sees the bread and incorporates it into the semantic spatial map. However, the *Navigator* does not return control to the *Planner*, as it has not yet reached the desired destination. Upon returning at timestep 45, the *Planner* reads the spatial map, sees the bread, and immediately decides it can answer the question. Thus π_{answer} and $P(True)$ both increase to nearly 100%. For more examples, please see our supplementary video https://youtu.be/pXd3C-1jr98.

6. Conclusion

In this work, we pose a new problem of Interactive Question Answering for several question types in interactive environments. We propose the Hierarchical Interactive Memory Network (HIMN) for this task, consisting of a factorized set of controllers, allowing the system to operate at multiple levels of temporal abstraction. We also introduce the Egocentric Spatial GRU for updating spatial memory maps. The effectiveness of our proposed model is demonstrated on a new benchmark dataset built upon a high-quality simulation environment for this task. This dataset still presents several challenges to our model and baselines and warrants future research.

7. Acknowledgements

This work was funded in part by the National Science Foundation under contract number NSF-NRI-1637479, ONR N00014-13-1-0720, the Allen Distinguished Investigator Award, and the Allen Institute for Artificial Intelligence. We would like to thank Xun Huang for initial discussions and dataset prototypes. We would also like to thank NVIDIA for generously providing a DGX used for this research via the UW NVIDIA AI Lab (NVAIL).

References

[1] Unity software. https://unity3d.com. 2

[2] A. Agrawal, J. Lu, S. Antol, M. Mitchell, C. L. Zitnick, D. Parikh, and D. Batra. Vqa: Visual question answering. *International Journal of Computer Vision*, 2017. 3

[3] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural module networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3

[4] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath. A brief survey of deep reinforcement learning. *CoRR*, abs/1708.05866, 2017. 3

[5] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *J. Artif. Intell. Res.(JAIR)*, 2013. 4

[6] J. Borenstein and Y. Koren. Real-time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, 1989. 3

[7] J. Borenstein and Y. Koren. The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE transactions on robotics and automation*, 1991. 3

[8] D. S. Chaplot, K. M. Sathyendra, R. K. Pasumarthi, D. Rajagopal, and R. Salakhutdinov. Gated-attention architectures for task-oriented language grounding. *CoRR*, abs/1706.07230, 2017. 2, 4, 7

[9] C. Chen, A. Seff, A. Kornhauser, and J. Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *ICCV*, 2015. 4

[10] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *ICCV*, 2003. 3

[11] T. G. Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. *J. Artif. Intell. Res.*, 2000. 3

[12] C. Dornhege, M. Gissler, M. Teschner, and B. Nebel. Integrating symbolic and geometric planning for mobile manipulation. In *Safety, Security & Rescue Robotics (SSRR), 2009 IEEE International Workshop on*, 2009. 4

[13] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *ECCV*, 2014. 3

[14] R. E. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 1971. 4

[15] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *EMNLP*, 2016. 3

[16] H. Gao, J. Mao, J. Zhou, Z. Huang, L. Wang, and W. Xu. Are you talking to a machine? dataset and methods for multilingual image question. In *NIPS*, 2015. 3

[17] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*, 2017. 5

[18] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik. Cognitive mapping and planning for visual navigation. *arXiv preprint arXiv:1702.03920*, 2017. 3

[19] H. Haddad, M. Khatib, S. Lacroix, and R. Chatila. Reactive navigation in outdoor environments using potential fields. In *International Conference on Robotics and Automation*, 1998. 3

[20] A. Handa, V. Ptrucean, V. Badrinarayanan, R. Cipolla, et al. Understanding realworld indoor sceneswith synthetic data. In *CVPR*, 2016. 4

[21] F. Hill, K. M. Hermann, P. Blunsom, and S. Clark. Understanding grounded language learning agents. *CoRR*, abs/1710.09867, 2017. 2, 4, 7

[22] R. Hu, J. Andreas, M. Rohrbach, T. Darrell, and K. Saenko. Learning to reason: End-to-end module networks for visual question answering. *CoRR*, abs/1704.05526, 2017. 3

[23] M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*, 2016. 9

[24] A. Jaegle, S. Phillips, and K. Daniilidis. Fast, robust, continuous monocular egomotion computation. In *International Conference on Robotics and Automation*, 2016. 3

[25] Y. Jang, Y. Song, Y. Yu, Y. Kim, and G. Kim. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. *CoRR*, abs/1704.04497, 2017. 3

[26] J. Johnson, B. Hariharan, L. van der Maaten, F. fei Li, C. L. Zitnick, and R. B. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. *CoRR*, abs/1612.06890, 2016. 3

[27] J. Johnson, B. Hariharan, L. van der Maaten, J. Hoffman, F. fei Li, C. L. Zitnick, and R. B. Girshick. Inferring and executing programs for visual reasoning. *CoRR*, abs/1705.03633, 2017. 3

[28] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *J. Artif. Intell. Res.*, 1996. 3

[29] L. P. Kaelbling and T. Lozano-Pérez. Hierarchical task and motion planning in the now. In *International Conference on Robotics and Automation*, 2011. 4

[30] S. E. Kahou, A. Atkinson, V. Michalski, Á. Kádár, A. Trischler, and Y. Bengio. Figureqa: An annotated figure dataset for visual reasoning. *CoRR*, abs/1710.07300, 2017. 3

[31] A. Kembhavi, M. Salvato, E. Kolve, M. J. Seo, H. Hajishirzi, and A. Farhadi. A diagram is worth a dozen images. In *ECCV*, 2016. 3

[32] A. Kembhavi, M. Seo, D. Schwenk, J. Choi, A. Farhadi, and H. Hajishirzi. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *CVPR*, 2017. 3

[33] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaśkowski. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *Computational Intelligence and Games (CIG), 2016 IEEE Conference on*, 2016. 4

[34] K. Kidono, J. Miura, and Y. Shirai. Autonomous visual navigation of a mobile robot using a human-guided experience. *Robotics and Autonomous Systems*, 2002. 3

[35] D. Kim and R. Nevatia. Symbolic navigation with a generic map. *Autonomous Robots*, 1999. 3

[36] K.-M. Kim, M.-O. Heo, S.-H. Choi, and B.-T. Zhang. Deepstory: Video story qa by deep embedded memory networks. In *IJCAI*, 2017. 3

[37] N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *International Conference on Robotics and Automation*, 2004. 3

[38] T. Kollar and N. Roy. Trajectory optimization using reinforcement learning for map exploration. *The International Journal of Robotics Research*, 2008. 3

[39] K. Konolige, J. Bowman, J. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. *The International Journal of Robotics Research*, 2010. 3

[40] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. S. Bernstein, and F. fei Li. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 2016. 3

[41] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *NIPS*, 2016. 3

[42] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, 2016. 6

[43] S. Lenser and M. Veloso. Visual sonar: Fast obstacle avoidance using monocular vision. In *International Conference on Intelligent Robots and Systems*, 2003. 3

[44] A. Lerer, S. Gross, and R. Fergus. Learning physical intuition of block towers by example. *arXiv preprint arXiv:1603.01312*, 2016. 4

[45] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 2016. 3

[46] Y. Liang, M. C. Machado, E. Talvitie, and M. Bowling. State of the art control of atari games using shallow reinforcement learning. In *International Conference on Autonomous Agents & Multiagent Systems*, 2016. 3

[47] C. Linegar, W. Churchill, and P. Newman. Made to measure: Bespoke landmarks for 24-hour, all-weather localisation with a camera. In *International Conference on Robotics and Automation*, 2016. 3

[48] J. Lu, J. Yang, D. Batra, and D. Parikh. Hierarchical question-image co-attention for visual question answering. In *NIPS*, 2016. 3

[49] M. Malinowski and M. Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *NIPS*, 2014. 3

[50] M. Malinowski, M. Rohrbach, and M. Fritz. Ask your neurons: A neural-based approach to answering questions about images. In *ICCV*, 2015. 3

[51] J. Marin, D. Vázquez, D. Gerónimo, and A. M. López. Learning appearance in virtual scenarios for pedestrian detection. In *CVPR*, 2010. 4

[52] C. McManus, B. Upcroft, and P. Newmann. Scene signatures: Localised and point-less features for localisation. 2014. 3

[53] J. Michels, A. Saxena, and A. Y. Ng. High speed obstacle avoidance using monocular vision and reinforcement learning. In *International Conference on Machine Learning*, 2005. 3

[54] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, 2016. 3, 6

[55] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 2015. 3

[56] R. Mottaghi, H. Bagherinezhad, M. Rastegari, and A. Farhadi. Newtonian scene understanding: Unfolding the dynamics of objects in static images. In *CVPR*, 2016. 4

[57] R. Mottaghi, M. Rastegari, A. Gupta, and A. Farhadi. what happens if... learning to predict the effect of forces in images. In *ECCV*, 2016. 4

[58] J. Mun, P. H. Seo, I. Jung, and B. Han. Marioqa: Answering questions by watching gameplay videos. *CoRR*, abs/1612.01669, 2016. 3

[59] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 2015. 3

[60] J. Oh, S. Singh, H. Lee, and P. Kohli. Communicating hierarchical neural controllers for learning zero-shot task generalization. 2016. 3

[61] G. Oriolo, M. Vendittelli, and G. Ulivi. On-line map building and navigation for autonomous mobile robots. In *International Conference on Robotics and Automation*, 1995. 3

[62] J. Papon and M. Schoeler. Semantic pose using deep networks trained on synthetic rgb-d. In *ICCV*, 2015. 4

[63] R. E. Parr and S. J. Russell. Reinforcement learning with hierarchies of machines. In *NIPS*, 1997. 3

[64] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 2008. 3

[65] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016. 6, 7, 9

[66] A. Remazeilles, F. Chaumette, and P. Gros. Robot motion control from a visual memory. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, 2004. 3

[67] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 4

[68] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016. 4

[69] E. Royer, J. Bom, M. Dhome, B. Thuilot, M. Lhuillier, and F. Marmoiton. Outdoor autonomous navigation using monocular vision. In *International Conference on Intelligent Robots and Systems*, 2005. 3

[70] F. Sadeghi and S. Levine. Cad2rl: Real single-image flight without a single real image. *CoRR*, abs/1611.04201, 2016. 3

[71] P. Saeedi, P. D. Lawrence, and D. G. Lowe. Vision-based 3-d trajectory tracking for unknown environments. *IEEE transactions on robotics*, 2006. 3

[72] A. Shafaei, J. J. Little, and M. Schmidt. Play and learn: Using video games to train computer vision models. *arXiv preprint arXiv:1608.01745*, 2016. 4

[73] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016. 3

[74] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. R. Baker, M. Lai, A. Bolton, Y. Chen, T. P. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis. Mastering the game of go without human knowledge. *Nature*, 2017. 3

[75] R. Sim and J. J. Little. Autonomous vision-based exploration and mapping using hybrid maps and rao-blackwellised particle filters. In *International Conference on Intelligent Robots and Systems*. IEEE, 2006. 3

[76] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel. Combined task and motion planning through an extensible planner-independent interface layer. In *International Conference on Robotics and Automation*, 2014. 4

[77] S. Srivastava, L. Riano, S. Russell, and P. Abbeel. Using classical planners for tasks with continuous operators in robotics. In *Intl. Conf. on Automated Planning and Scheduling*, 2013. 4

[78] R. S. Sutton, D. Precup, and S. P. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.*, 1999. 3

[79] M. Tapaswi, Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun, and S. Fidler. Movieqa: Understanding stories in movies through question-answering. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4631–4640, 2016. 3

[80] C. Tessler, S. Givony, T. Zahavy, D. J. Mankowitz, and S. Mannor. A deep hierarchical approach to lifelong learning in minecraft. In *AAAI*, 2017. 3

[81] M. Tomono. 3-d object map building using dense object models with sift-based recognition features. In *International Conference on Intelligent Robots and Systems*, 2006. 3

[82] K. Tu, M. Meng, M. W. Lee, T. E. Choe, and S.-C. Zhu. Joint video and text parsing for understanding events and answering queries. *IEEE MultiMedia*, 2014. 3

[83] P. Wang, Q. Wu, C. Shen, A. R. Dick, and A. van den Hengel. Explicit knowledge-based reasoning for visual question answering. In *IJCAI*, 2017. 3

[84] P. Wang, Q. Wu, C. Shen, A. van den Hengel, and A. R. Dick. Fvqa: Fact-based visual question answering. *IEEE transactions on pattern analysis and machine intelligence*, 2017. 3

[85] D. Wooden. A guide to vision-based map building. *IEEE Robotics & Automation Magazine*, 2006. 3

[86] J. Wu, I. Yildirim, J. J. Lim, B. Freeman, and J. Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *NIPS*, 2015. 4

[87] Q. Wu, D. Teney, P. Wang, C. Shen, A. R. Dick, and A. van den Hengel. Visual question answering: A survey of methods and datasets. *CoRR*, abs/1607.05910, 2016. 3

[88] Q. Wu, P. Wang, C. Shen, A. R. Dick, and A. van den Hengel. Ask me anything: Free-form visual question answering based on knowledge from external sources. *CVPR*, 2016. 3

[89] B. Wymann, E. Espié, C. Guionneau, C. Dimitrakakis, R. Coulom, and A. Sumner. Torcs, the open racing car simulator. *Software available at http://torcs. sourceforge. net*, 2000. 4

[90] C. Xiong, S. Merity, and R. Socher. Dynamic memory networks for visual and textual question answering. In *ICML*, 2016. 3

[91] H. Xu and K. Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *ECCV*, 2016. 3

[92] Z. Yang, X. He, J. Gao, L. Deng, and A. J. Smola. Stacked attention networks for image question answering. *CVPR*, 2016. 3

[93] Y. Zhu, D. Gordon, E. Kolve, D. Fox, L. Fei-Fei, A. Gupta, R. Mottaghi, and A. Farhadi. Visual semantic planning using deep successor representations. *Proceedings of the IEEE International Conference on Computer Vision*, 2017. 2, 4, 7, 9

[94] Y. Zhu, O. Groth, M. S. Bernstein, and F. fei Li. Visual7w: Grounded question answering in images. *CVPR*, 2016. 3

[95] 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 *International Conference on Robotics and Automation*, 2017. 1, 2, 3, 4, 9