A Systematic Survey of Text Worlds as Embodied Natural Language Environments

Peter A. Jansen
University of Arizona, Tucson, AZ
pajansen@arizona.edu

Abstract

Text Worlds are virtual environments for embodied agents that, unlike 2D or 3D environments, are rendered exclusively using textual descriptions. These environments offer an alternative to higher-fidelity 3D environments due to their low barrier to entry, providing the ability to study semantics, compositional inference, and other high-level tasks with rich action spaces while controlling for perceptual input. This systematic survey outlines recent developments in tooling, environments, and agent modeling for Text Worlds, while examining recent trends in knowledge graphs, common sense reasoning, transfer learning of Text World performance to higher-fidelity environments, as well as near-term development targets that, once achieved, make Text Worlds an attractive general research paradigm for natural language processing.

1 Introduction

Embodied agents offer an experimental paradigm to study the development and use of semantic representations for a variety of real-world tasks, from household tasks (Shridhar et al., 2020a) to navigation (Guss et al., 2019) to chemical synthesis (Tamari et al., 2021). While robotic agents are a primary vehicle for studying embodiment (e.g. Cangelosi and Schlesinger, 2015), robotic models are costly to construct, and experiments can be slow or difficult to scale. Virtual agents and embodied virtual environments help mitigate many of these issues, allowing large-scale simulations to be run in parallel orders of magnitude faster than real world environments (e.g. Deitke et al., 2020), while controlled virtual environments can be constructed for exploring specific tasks – though this benefit in speed comes at the cost of having to model virtual 3D environments, which can be substantial.

Text Worlds – embodied environments rendered linguistically through textual descriptions instead of graphically through pixels (see Table 1) – have emerged as a recent methodological focus that allow studying many embodied research questions while reducing some of the development costs associated with modeling complex and photorealistic 3D environments (e.g. Côté et al., 2018). More than simply reducing development costs, Text Worlds also offer paradigms to study developmental knowledge representation, embodied task learning, and transfer learning at a higher level than perceptually-grounded studies, enabling different research questions that explore these topics in isolation of the open problems of perceptual input, object segmentation, and object classification regularly studied in the vision community (e.g. He et al., 2016c; Szegedy et al., 2017; Zhai et al., 2021).

Zork
North of House You are facing the north side of a white house. There is no door here, and all the windows are barred. >*go north*
Forest This is a dimly lit forest, with large trees all around. One particularly large tree with some low branches stands here. >*climb large tree*
Up a Tree You are about 10 feet above the ground nestled among some large branches. On the branch is a small birds nest. In the bird's nest is a large egg encrusted with precious jewels, apparently scavenged somewhere by a childless songbird. >*take egg*
Taken. >*climb down tree*
Forest >*go north*

Table 1: An example Text World interactive fiction environment, Zork (Lebling et al., 1979), frequently used as a benchmark for agent performance. User-entered actions are *italicized*.

1.1 Motivation for this survey

Text Worlds are rapidly gaining momentum as a research methodology in the natural language processing community. In spite of this interest, many modeling, evaluation, tooling, and other barriers exist to applying these methodologies, with significant development efforts in the early stages of

mitigating those barriers, at least in part.

In this review, citation graphs of recent articles were iteratively crawled, identifying 108 articles relevant to Text Worlds and other embodied environments that include text as part of the simulation or task. Frequent motivations for choosing Text Worlds are highlighted in Section 2. Tooling and modeling paradigms (in the form of simulators, intermediate languages, and libraries) are surveyed in Section 3, with text environments and common benchmarks implemented with this tooling described in Section 4. Contemporary focuses in agent modeling, including coupling knowledge graphs, question answering, and common-sense reasoning with reinforcement learning, are identified in Section 5. Recent contributions to focus areas in world generation and hybrid text-3D environments are summarized in Section 6, while a distillation of near-term directions for reducing barriers to using Text Worlds more broadly as a research paradigm are presented in Section 7.

2 Why use Text Worlds?

For many tasks, Text Worlds can offer advantages over other embodied environment modelling paradigms – typically in reduced development costs, the ability to model large action spaces, and the ability to study embodied reasoning at a higher level than raw perceptual information.

Embodied Reasoning: Embodied agents have been proposed as a solution to the symbol grounding problem (Harnad, 1990), or the problem of how concepts acquire real-world meaning. Humans likely resolve symbol grounding at least partially by assigning semantics to concepts through perceptually-grounded mental simulations (Barsalou et al., 1999). Using embodied agents that take in perceptual data and perform actions in real or virtual environments offers an avenue for studying semantics and symbol grounding empirically (Cangelosi et al., 2010; Bisk et al., 2020; Tamari et al., 2020a,b). Text Worlds abstract some of the challenges in perceptual modeling, allowing agents to focus on higher-level semantics, while hybrid worlds that simultaneously render both text and 3D views (e.g. Shridhar et al., 2020b) help control what kind of knowledge is acquired, and better operationalize the study of symbol grounding.

Ease of Development: Constructing embodied virtual environments typically has steep development costs, but Text Worlds are typically easier to construct for many tasks. Creating new objects does not require the expensive process of creating new 3D models, or performing visual-percept-to-object-name segmentation or classification (since the scene is rendered linguistically). Similarly, a rich action semantics is possible, and comparatively easy to implement – while 3D environments typically have one or a small number of action commands (e.g. Kolve et al., 2017; Shridhar et al., 2020a), Text Worlds typically implement dozens of action verbs, and thousands of valid *Verb-NounPhrase* action combinations (Hausknecht et al., 2020).

Compositional Reasoning: Complex reasoning tasks typically require multi-step (or compositional) reasoning that integrates several pieces of knowledge in an action procedure that arrives at a solution. In the context of natural language, compositional reasoning is frequently studied through question answering tasks (e.g. Yang et al., 2018; Khot et al., 2020; Xie et al., 2020; Dalvi et al., 2021) or procedural knowledge prediction (e.g. Dalvi et al., 2018; Tandon et al., 2018; Dalvi et al., 2019). A contemporary challenge is that the number of valid compositional procedures is typically large compared to those that can be tractably annotated as gold, and as such automatically evaluating model performance becomes challenging (Jansen et al., 2021). In an embodied environment, an agent's actions have (generally) deterministic consequences for a given environment state, as actions are grounded in an underlying action language (e.g. McDermott et al., 1998) or linear logic (e.g. Martens, 2015). Embodied environments can offer a more formal semantics to study these reasoning tasks, where correctness of novel procedures could be evaluated directly.

Transfer Learning: Training a text-only agent for embodied tasks allows the agent to learn those tasks in a distilled form, at a high-level. This performance can then be transferred to more realistic 3D environments, where agents pretrained on text versions of the same environment learn to ground their high-level knowledge in low-level perceptual information, and complete tasks faster than when trained jointly (Shridhar et al., 2020b). This offers the possibility of creating simplified text worlds to pretrain agents for challenging 3D tasks that are currently out of reach of embodied agents.

3 Text World Simulators

Text World simulators render an agent's world view directly into textual descriptions of their environment, rather than into 2D or 3D graphical renderings. Similarly, actions the agent wishes to take are provided to the simulator as text (e.g. *"read the letter"* in *Zork*), requiring agent models to both parse input text from the environment, and generate output text to to interact with that environment.

In terms of simulators, the Z-machine (Infocom, 1989) is a low-level virtual machine originally designed by Infocom for creating portable interactive fiction novels (such as *Zork*). It was paired with a high-level LISP-like domain-specific language (ZIL) that included libraries for text parsing, and other tools for writing interactive fiction novels. The Z-machine standard was reverse-engineered by others (e.g. Nelson, 2014) in an effort to build their own high-level interactive fiction domain-specific languages, and has since become a standard compilation target due to the proliferation of existing tooling and legacy environments.[1]

Inform7 (Nelson, 2006) is a popular high-level language designed for interactive fiction novels that allows environment rules to be directly specified in a simplified natural language, substantially lowering the barrier to entry for creating text worlds. The text generation engine allows substantial variation in the way the environments are described, from dry formulaic text to more natural, varied, conversational descriptions. Inform7 is compiled to Inform6, an earlier object-oriented scripting language with C-like syntax, which itself is compiled to Z-machine code.

Ceptre (Martens, 2015) is a linear-logic simulation engine developed with the goal of specifying more generic tooling for operational logics than Inform 7. TextWorld (Côté et al., 2018) adapt Ceptre's linear logic state transitions for environment descriptions, and add tooling for generative environments, visualization, and RL agent coupling, all of which is compiled into Inform7 source code. Parallel to this, the Jericho environment (Hausknecht et al., 2020) allows inferring relevant vocabulary and template-based object interactions for Z-machine-based interactive fiction games, easing action selection for agents.

[1] A variety of text adventure tooling, including the Adventure Game Toolkit (AGT) and Text Adventure Development System (TADS), was developed starting in the late 1980s, but these simulators have generally not been adopted by the NLP community in favour of the more popular Inform series tools.

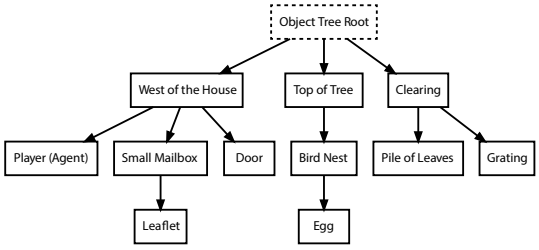


Figure 1: An example partial object tree from the interactive fiction game *Zork* (Lebling et al., 1979).

3.1 Text World Modeling Paradigms

3.1.1 Environment Modelling

Environments are typically modeled as an **object tree** that represents all the objects in an environment and their nested locations, as well as a set of **action rules** that implement changes to the objects in the environment based on an agent's actions.

Objects: Because of the body of existing interactive fiction environments for Z-machine environments, and nearly all popular tooling (Inform7, TextWorlds, etc.) ultimately compiling to Z-machine code, object models typically use the Z-machine model (Nelson, 2014). Z-machine objects have **names** (e.g. "mailbox"), **descriptions** (e.g. "a small wooden mailbox"), binary flags called **attributes** (e.g. "is_container_open"), and generic **properties** stored as key-value pairs. Objects are stored in the *object tree*, which represents the locations of all objects in the environment through parent-child relationships, as shown in Figure 1.

Action Rules: Action rules describe how objects change in response to a given world **state**, which is frequently a collection of preconditions followed by an action taken by an agent (e.g. *"eat the apple"*), but can also be due to environment states (e.g. a plant dying because it hasn't been watered for a time greater than some threshold). Ceptre (Martens, 2015) and TextWorld (Côté et al., 2018) use **linear logic** to represent possible valid **state transitions**. In linear logic, a set of *preconditions* in the state history of the world can be consumed by a rule to generate a set of *postconditions*, such as consuming a `closed(C)` precondition and posting a `open(C)` postcondition for a container-opening action for some container `C`.

Côté et al. (2018) note the limitations in existing implementations of state transition systems for text worlds (such as single-step forward or backward chaining), and suggest future systems may wish

to use mature **action languages** such as STRIPS (Fikes and Nilsson, 1971) or GDL (Genesereth et al., 2005; Thielscher, 2010, 2017) as the basis of a world model, though each of these languages have tradeoffs in features (such as object typing) and general expressivity (such as being primarily agent-action centered, rather than implementing environment-driven actions and processes) that make certain kinds of complex modeling more challenging. As a proof-of-concept, ALFWorld (Shridhar et al., 2020b) uses the Planning Domain Definition Language (PDDL, McDermott et al., 1998) to define the semantics for the variety of *pick-and-place* tasks in its text world rendering of the ALFRED benchmark.

3.1.2 Agent Modelling

While environments can be modelled as a collection of states and allowable state transitions (or rules), agents typically have incomplete or inaccurate information about the environment, and must make **observations** of the environment state through (potentially noisy or inadequate) sensors, and take **actions** based on those observations. Because of this, agents are typically modelled as partially-observable Markov decision processes (POMDP) (Kaelbling et al., 1998).

A Markov decision process (MDP) contains the state history (S), valid state transitions (T), available actions (A), and (for agent modeling) the expected immediate reward for taking each action (R). POMDPs extend this to account for partial observability by supplying a finite list of observations the agent can make (Ω), and an observation function (O) that returns what the agent actually observes from an observation, given the current world state. For example, the observation function might return *unknown* if the agent tries to examine the contents of a locked container before unlocking it, because the contents cannot yet be observed. Similarly, when observing the temperature of a cup of tea, the observation function might return coarse measurements *(e.g. hot, warm, cool)* if the agent uses their hand for measurement, or fine-grained measurements *(e.g.* $70°C$*)* if the agent uses a thermometer. A final discount factor (γ) influences whether the agent prefers immediate rewards, or eventual (distant) rewards. The POMDP defined by defined by $(S, T, A, R, \Omega, O, \gamma)$ then serves as a model for a learning framework, typically reinforcement learning (RL), to learn a policy that enables the agent to maximize the reward.

4 Text World Environments

Environments are worlds implemented in simulators, that agents explore to perform tasks. Environments can be simple or complex, evaluate task-specific or domain-general competencies, be static or generative, and have small or large action spaces compared to higher-fidelity simulators (see the Appendix for a comparison of action space sizes across environments and simulators).

4.1 Single Environment Benchmarks

Single environment benchmarks typically consist of small environments designed to test specific agent competencies, or larger interactive fiction environments that test broad agent competencies to navigate a large world and interact with the environment toward achieving some distant goal. Toy environments frequently evaluate an agent's ability to perform compositional reasoning tasks of increasing lengths, such as in the Kitchen Cleanup and related benchmarks (Murugesan et al., 2020b). Other toy worlds explore searching environments to locate specific objects (Yuan et al., 2018), or combining source materials to form new materials (Jiang et al., 2020). While collections of interactive fiction environments are used as benchmarks (see Section 4.3), individual environments frequently form single benchmarks. Zork (Lebling et al., 1979) and its subquests are medium-difficulty environments frequently used in this capacity, while Anchorhead (Gentry, 1998) is a challenging environment where state-of-the-art performance remains below 1%.

4.2 Domain-specific Environments

Domain-specific environments allow agents to learn highly specific competencies relevant to a single domain, like science or medicine, while typically involving more modeling depth than toy environments. Tamari et al. (2021) create a TextWorld environment for wet lab chemistry protocols, that describe detailed step-by-step instructions for replicating chemistry experiments. These text-based simulations can then be represented as *process execution graphs (PEG)*, which can then be run on real lab equipment. A similar environment exists for the materials science domain (Tamari et al., 2019).

4.3 Environment Collections as Benchmarks

To test the generality of agents, large collections of interactive fiction games (rather than single environments) are frequently used as benchmarks. While

the Text-Based Adventure AI Shared Task initially evaluated on a single benchmark environment, later instances switched to evaluating on 20 varied environments to gauge generalization (Atkinson et al., 2019). Fulda et al. (2017a) created a list of 50 interactive fiction games to serve as a benchmark for agents to learn common-sense reasoning. Côté et al. (2018) further curate this list, replacing 20 games without scores to those more useful for RL agents. The Jericho benchmark (Hausknecht et al., 2020) includes 32 interactive fiction games that support Jericho's in-built methods for score and world-change detection, out of a total of 56 games known to support these features.

4.4 Generative Environments

A difficulty with statically-initialized environments is that because their structure is identical each time the simulation is run, rather than learning general skills, agents quickly overfit to a particular task and environment, and rarely generalize to unseen environments (Chaudhury et al., 2020). Procedurally generated environments help address this need by generating variations of environments centered around specific goal conditions.

The TextWorld simulator (Côté et al., 2018) allows specifying high-level parameters such as the number of rooms, objects, and winning conditions, then uses a random walk to procedurally generate environment maps in the Inform7 language meeting those specifications, using either forward or backward chaining during generation to verify tasks can be successfully completed in the random environment. As an example, the First TextWorld Problems shared task[2] used TextWorld to generate 5k variations of a cooking environment, divided into train, development, and test sets. Similarly, Murugesan et al. (2020a) introduce TextWorld CommonSense (TWC), a simple generative environment for household cleaning tasks, modelled as a *pick-and-place* task where agents must pick up common objects from the floor, and place them in their common household locations (such as placing *shoes* in a *shoe cabinet*). Other related environments include Coin Collector (Yuan et al., 2018), a generative environment for a navigation task, and Yin et al.'s (2019b) procedurally generated environment for cooking tasks.

Adhikari et al. (2020) generate a large set of recipe-based cooking games, where an agent must precisely follow a cooking recipe that requires collecting tools (e.g. *a knife*) and ingredients (e.g. *carrots*), and processing those ingredients correctly (e.g. *dice carrots, cook carrots*) in the correct order. Jain et al. (2020) propose a similar synthetic benchmark for multi-step compositional reasoning called SaladWorld. In the context of question answering, Yuan et al. (2019) procedurally generate a simple environment that requires an agent to search and investigate attributes of objects, such as verifying their existence, locations, or specific attributes (like edibility). On the balance, while tooling exists to generate simple procedural environments, when compared to classic interactive fiction games (such as *Zork*), the current state-of-the-art allows for generating only relatively simple environments with comparatively simple tasks and near-term goals than human-authored interactive fiction games.

5 Text World Agents

Recently a large number of agents have been proposed for Text World environments. This section briefly surveys common modeling methods, paradigms, and trends, with the performance of recent agents on common interactive fiction games (as categorized by the Jericho benchmark, Hausknecht et al., 2020) shown in Table 2.

Reinforcement Learning: While some agents rely on learning frameworks heavily coupled with heuristics (e.g., Kostka et al., 2017, Golovin), owing to the sampling benefits afforded by operating in a virtual environment, the predominant modeling paradigm for most contemporary text world agents is reinforcement learning. Narasimhan et al. (2015) demonstrate that "Deep-Q Networks" (DQN) (Mnih et al., 2015) developed for Atari games can be augmented with LSTMs for representation learning in Text Worlds, which outperform simpler methods using n-gram bag-of-words representations. He et al. (2016a, DRRN) extend this to build the Deep Reinforcement Relevance Network (DRRN), an architecture that uses separate embeddings for the state space and actions, to improve both training time and performance. Madotto et al. (2020) show that the Go-Explore algorithm (Ecoffet et al., 2019), which periodically returns to promising but underexplored areas of a world, can achieve higher scores than the DRRN with fewer steps. Zahvey et al. (2018, AE-DQN) use an Action Elimination Network (AEN) to remove sub-

[2] https://competitions.codalab.org/competitions/21557

Model	Detective (E)	Zork1 (M)	Zork3 (M)	OmniQuest (M)	Spirit (H)	Enchanter (H)
DRRN (He et al., 2016b)	0.55	0.09	0.07	0.20	0.05	0.00
BYU-Agent (Fulda et al., 2017a)	0.59	0.03	0.00	0.10	0.00	0.01
Golovin (Kostka et al., 2017)	0.20	0.04	0.10	0.15	0.00	0.01
AE-DQN (Zahavy et al., 2018)	–	0.05	–	–	–	–
NeuroAgent (Rajalingam and Samothrakis, 2019)	0.19	0.03	0.00	0.20	0.00	0.00
NAIL (Hausknecht et al., 2019)	0.38	0.03	0.26	–	0.00	0.00
CNN-DQN (Yin and May, 2019a)	–	0.11	–	–	–	–
IK-OMP (Tessler et al., 2019)	–	1.00	–	–	–	–
TDQN (Hausknecht et al., 2020)	0.47	0.03	0.00	0.34	0.02	0.00
KG-A2C (Ammanabrolu and Hausknecht, 2020)	0.58	0.10	0.01	0.06	0.03	0.01
SC (Jain et al., 2020)	–	0.10	–	–	0.0	–
CALM (N-gram) (Yao et al., 2020)	0.79	0.07	0.00	0.09	0.00	0.00
CALM (GPT-2) (Yao et al., 2020)	0.80	0.09	0.07	0.14	0.05	0.01
RC-DQN (Guo et al., 2020a)	0.81	0.11	0.40	0.20	0.05	0.02
MPRC-DQN (Guo et al., 2020a)	0.88	0.11	0.52	0.20	0.05	0.02
SHA-KG (Xu et al., 2020)	0.86	0.10	0.10	–	0.05	0.02
MC!Q*BERT (Ammanabrolu et al., 2020b)	0.92	0.12	–	–	0.00	–
INV-DY (Yao et al., 2021)	0.81	0.12	0.06	0.11	0.05	–

Table 2: Agent performance on benchmark interactive fiction environments. All performance values are normalized to maximum achievable scores in a given environment. Due to the lack of standard reporting practice, performance reflects values reported for agents, but is unable to hold other elements (such as number of training epochs, number of testing epochs, reporting average vs maximum performance) constant. Parentheses denote environment difficulty (E:Easy, M:Medium, H:Hard) as determined by the Jericho benchmark (Hausknecht et al., 2020).

optimal actions, showing improved performance over a DQN on *Zork*. Yao et al (2020, CALM) use a GPT-2 language model trained on human gameplay to reduce the space of possible input command sequences, and produce a shortlist of candidate actions for an RL agent to select from. Yao et al. (2021, INV-DY) demonstrate that semantic modeling is important, showing that models that either encode semantics through an inverse dynamic decoder, or discard semantics by replacing words with unique hashes, have different performance distributions in different environments. Taking a different approach, Tessler et al. (2019, IK-OMP) show that imitation learning combined with a compressed sensing framework can solve Zork when restricted to a vocabulary of 112 words extracted from walk-through examples.

Constructing Graphs: Augmenting reinforcement learning models to produce knowledge graphs of their beliefs can reduce training time and improve overall agent performance (Ammanabrolu and Riedl, 2019). Ammanabrolu et al. (2020, KG-A2C) demonstrate a method for training an RL agent that uses a knowledge graph to model its state-space, and use a template-based action space to achieve strong performance across a variety of interactive fiction benchmarks. Adhikari et al. (2020) demonstrate that a Graph Aided Transformer Agent (GATA) is able to learn implicit belief networks about its environment, improving agent performance in a cooking environment. Xu et al. (2020, SHA-KG) extend KG-A2C to use hierarchical RL to reason over subgraphs, showing substantially improved performance on a variety of benchmarks.

To support these modelling paradigms, Zelinka et al. (2019) introduce TextWorld KG, a dataset for learning the subtask of updating knowledge graphs based on text world descriptions in a cooking domain, and show their best ensemble model is able to achieve 70 F1 at this subtask. Similarly, Annamabrolu et al. (2021a) introduce JerichoWorld, a similar dataset for world modeling using knowledge graphs but on a broader set of interactive fiction games, and subsequently introduce WorldFormer (Ammanabrolu and Riedl, 2021b), a multi-task transformer model that performs well at both knowledge-graph prediction and next-action prediction tasks.

Question Answering: Agents can reframe Text World tasks as question answering tasks to gain relevant knowledge for action selection, with these agents providing current state-of-the-art performance across a variety of benchmarks. Guo et al. (2020b, MPRC-DQN) use multi-paragraph reading comprehension (MPRC) techniques to ask questions that populate action templates for agents, substantially reducing the number of training examples required for RL agents while achieving strong per-

formance on the Jericho benchmark. Similarly, Ammanabrolu et al. (2020b, MC!Q*BERT) use contextually-relevant questions (such as *"Where am I?", "Why am I here?"*) to populate their knowledge base to support task completion.

Common-sense Reasoning: Agents arguably require a large background of common-sense or world knowledge to perform embodied reasoning in virtual environments. Fulda et al. (2017a) extract common-sense affordances from word vectors trained on Wikipedia using word2vec (Mikolov et al., 2013), and use this to increase performance on interactive fiction games, as well as (more generally) on robotic learning tasks (Fulda et al., 2017b). Murugesan et al. (2020b) combine the ConceptNet common-sense knowledge graph (Speer et al., 2017) with an RL agent that segments knowledge between general world knowledge, and specific beliefs about the current environment, demonstrating improved performance in a cooking environment. Similarly, Dambekodi et al. (2020) demonstrate that RL agents augmented with either COMET (Bosselut et al., 2019), a transformer trained on common-sense knowledge bases, or BERT (Devlin et al., 2019), which is hypothesized to contain common-sense knowledge, outperform agents without this knowledge on the interactive fiction game *9:05*. In the context of social reasoning, Ammanabrolu et al. (2021) create a fantasy-themed knowledge graph, ATOMIC-LIGHT, and show that an RL agent using this knowledge base performs well at the LIGHT social reasoning tasks.

6 Contemporary Focus Areas

World Generation: Generating detailed environments with complex tasks is labourious, while randomly generating environments currently provides limited task complexity and environment cohesiveness. World generation aims to support the generation of complex, coherent environments, either through better tooling for human authors (e.g. Temprado-Battad et al., 2019), or automated generation systems that may or may not have a human-in-the-loop. Fan et al. (2020) explore creating cohesive game worlds in the LIGHT environment using a variety of embedding models including Starspace (Wu et al., 2018a) and BERT (Devlin et al., 2019). Automatic evaluations show performance of between 36-47% in world building, defined as cohesively populating an environment with locations, objects, and characters. Similarly, human evaluation shows that users prefer Starspace-generated environments over those generated by a random baseline. In a more restricted domain, Ammanabrolu et al. (2019) show that two models, one Markov chain model, the other a generative language model (GPT-2), are capable of generating quests in a cooking environment, while there is a tradeoff between human ratings of quest creativity and coherence.

Ammanabrolu et al. (2020a) propose a large-scale end-to-end solution to world generation that automatically constructs interactive fiction environments based on a story (such as *Sherlock Holmes*) provided as input. Their system first builds a knowledge graph of the story by framing KG construction as a question answering task, using their model (AskBERT) to populate this graph. The system then uses either a rule-based baseline or a generative model (GPT-2) to generate textual descriptions of the world from this knowledge graph. User studies show that humans generally prefer these neural-generated worlds to the rule-generated worlds (measured in terms of interest, coherence, and genre-resemblance), but that neural-generated performance still substantially lags behind that of human-generated worlds.

Hybrid 3D-Text Environments: Hybrid simulators that can simultaneously render worlds both graphically (2D or 3D) as well as textually offer a mechanism to quickly learn high-level tasks without having to first solve grounding or perceptual learning challenges. The ALFWorld simulator (Shridhar et al., 2020b) combines the ALFRED 3D home environment (Shridhar et al., 2020a) with a simultaneous TextWorld interface to that same environment, and introduce the BUTLER agent, which shows increased task generalization on the 3D environment when first trained on the text world. Prior to ALFWorld, Jansen (2020) showed that a language model (GPT-2) was able to successfully generate detailed step-by-step textual descriptions of ALFRED task trajectories for up to 58% of unseen cases using task descriptions alone, without visual input. Building on this, Micheli (2021) confirmed GPT-2 also performs well on the text world rendering of ALFWorld, and is able to successfully complete goals in 95% of unseen cases. Taken together, these results show the promise of quickly learning complex tasks at a high-level in a text-only environment, then transferring this performance to agents grounded in more complex environments.

7 Contemporary Limitations and Challenges

Environment complexity is limited, and it's currently difficult to author complex worlds. Two competing needs are currently at odds: the desire for complex environments to learn complex skills, and the desire for environment variation to encourage robustness in models. Current tooling emphasizes creating varied procedural environments, but those environments have limited complexity, and require agents to complete straightforward tasks. Economically creating complex, interactive environments that simulate a significant fraction of real world interactions is still well beyond current simulators or libraries – but required for higher-fidelity interactive worlds that have multiple meaningful paths toward achieving task goals. Generating these environments semi-automatically (e.g. Ammanabrolu et al., 2020a) may offer a partial solution. Independent of tooling, libraries and other middleware offer near-term solutions to more complex environment modeling, much in the same way 3D game engines are regularly coupled with physics engine middleware to dramatically reduce the time required to implement forces, collisions, lighting, and other physics-based modeling. Currently, few analogs exist for text worlds. The addition of a *chemistry engine* that knows ice warmed above the freezing point will change to liquid water, or a *generator engine* that knows the sun is a source of sunlight during sunny days, or an *observation engine* that knows tools (like microscopes or thermometers) can change the observation model of a POMDP – may offer tractability in the form of modularization. Efforts using large-scale crowdsourcing to construct knowledge bases of commonsense knowledge (e.g., ATOMIC, Sap et al., 2019) may be required to support these efforts.

Current planning languages offer a partial solution for environment modelling. While simulators partially implement facilities for world modeling, some (e.g. Côté et al., 2018; Shridhar et al., 2020b) suggest using mature planning languages like STRIPS (Fikes and Nilsson, 1971) or PDDL (McDermott et al., 1998) for more full-featured modeling. This would not be without significant development effort – existing implementations of planning languages typically assume full-world observability (in conflict with POMDP modelling), and primarily agent-directed state-space changes, making complex world modeling with partial observability, and complex environment processes (such as plants that require water and light to survive, or a sun that rises and sets causing different items to be observable in day versus night) outside the space of being easily implemented with off-the-shelf solutions. In the near-term, it is likely that a domain-specific language specific to complex text world modeling would be required to address these needs while simultaneously reducing the time investment and barrier-to-entry for end users.

Analyses of environment complexity can inform agent design and evaluation. Text world articles frequently emphasize agent modeling contributions over environment, methodological, or analysis contributions – but these contributions are critical, especially in the early stages of this subfield. Agent performance in easy environments has increased incrementally, while medium-to-hard environments have seen comparatively modest improvements. Agent performance is typically reported as a distribution over a large number of environments, and the methodological groundwork required to understand when different models exceed others in time or performance over these environment distributions is critical to making forward progress. Transfer learning in the form of training on one set of environments and testing on others has become a standard feature of benchmarks (e.g. Hausknecht et al., 2020), but focused contributions that work to precisely characterize the limits of what can be learned from (for example) *OmniQuest* and transferred to *Zork*, and what capacities must be learned elsewhere, will help inform research programs in agent modeling and environment design.

Transfer learning between text world and 3D environments. Tasks learned at a high-level in text worlds help speed learning when those same models are transferred to more complex 3D environments (Shridhar et al., 2020b). This framing of transfer learning may resemble how humans can converse about plans for future actions in locations remote from those eventual actions (as when we apply knowledge learned in classrooms to the real world). As such, text-plus-3D environment rendering shows promise as a manner of controlling for different sources of complexity in multi-modal task learning (from high-level task-specific knowledge to low-level perceptual knowledge), and appears a promising research methodology for imparting complex task knowledge on agents that are able to navigate high-fidelity virtual environments.

References

Ashutosh Adhikari, Xingdi Yuan, Marc-Alexandre Côté, Mikuláš Zelinka, Marc-Antoine Rondeau, Romain Laroche, Pascal Poupart, Jian Tang, Adam Trischler, and Will Hamilton. 2020. Learning dynamic belief graphs to generalize on text-based games. *Advances in Neural Information Processing Systems*, 33.

Prithviraj Ammanabrolu, William Broniec, Alex Mueller, Jeremy Paul, and Mark Riedl. 2019. Toward automated quest generation in text-adventure games. In *Proceedings of the 4th Workshop on Computational Creativity in Language Generation*, pages 1–12.

Prithviraj Ammanabrolu, Wesley Cheung, Dan Tu, William Broniec, and Mark Riedl. 2020a. Bringing stories alive: Generating interactive fiction worlds. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 16, pages 3–9.

Prithviraj Ammanabrolu and Matthew Hausknecht. 2020. Graph constrained reinforcement learning for natural language action spaces. *arXiv preprint arXiv:2001.08837*.

Prithviraj Ammanabrolu and Mark Riedl. 2019. Playing text-adventure games with graph-based deep reinforcement learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3557–3565, Minneapolis, Minnesota. Association for Computational Linguistics.

Prithviraj Ammanabrolu and Mark Riedl. 2021a. Modeling worlds in text. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Prithviraj Ammanabrolu and Mark O Riedl. 2021b. Learning knowledge graph-based world models of textual environments. *arXiv preprint arXiv:2106.09608*.

Prithviraj Ammanabrolu, Ethan Tien, Matthew Hausknecht, and Mark O Riedl. 2020b. How to avoid being eaten by a grue: Structured exploration strategies for textual worlds. *arXiv preprint arXiv:2006.07409*.

Prithviraj Ammanabrolu, Jack Urbanek, Margaret Li, Arthur Szlam, Tim Rocktäschel, and Jason Weston. 2021. How to motivate your dragon: Teaching goal-driven agents to speak and act in fantasy worlds. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 807–833, Online. Association for Computational Linguistics.

Andrea Asperti, Carlo De Pieri, and Gianmaria Pedrini. 2017. Rogueinabox: an environment for roguelike learning. *International Journal of Computers*, 2.

Timothy Atkinson, Hendrik Baier, Tara Copplestone, Sam Devlin, and Jerry Swan. 2019. The text-based adventure ai competition. *IEEE Transactions on Games*, 11(3):260–266.

Chris Bamford. 2021. Griddly: A platform for ai research in games. *Software Impacts*, 8:100066.

Lawrence W Barsalou et al. 1999. Perceptual symbol systems. *Behavioral and brain sciences*, 22(4):577–660.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, et al. 2020. Experience grounds language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8718–8735.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779.

Angelo Cangelosi, Giorgio Metta, Gerhard Sagerer, Stefano Nolfi, Chrystopher Nehaniv, Kerstin Fischer, Jun Tani, Tony Belpaeme, Giulio Sandini, Francesco Nori, et al. 2010. Integration of action and language knowledge: A roadmap for developmental robotics. *IEEE Transactions on Autonomous Mental Development*, 2(3):167–195.

Angelo Cangelosi and Matthew Schlesinger. 2015. *Developmental Robotics: From Babies to Robots*. The MIT Press.

Thomas Carta, Subhajit Chaudhury, Kartik Talamadupula, and Michiaki Tatsubori. 2020. Visualhints: A visual-lingual environment for multimodal reinforcement learning. *arXiv preprint arXiv:2010.13839*.

Subhajit Chaudhury, Daiki Kimura, Kartik Talamadupula, Michiaki Tatsubori, Asim Munawar, and Ryuki Tachibana. 2020. Bootstrapped Q-learning with context relevant observation pruning to generalize in text-based games. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3002–3008, Online. Association for Computational Linguistics.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. 2018. Babyai: A platform to study the sample efficiency of grounded language learning. *arXiv preprint arXiv:1810.08272*.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. 2018. Textworld: A learning environment for text-based games. In *Workshop on Computer Games*, pages 41–75. Springer.

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1595–1604, New Orleans, Louisiana. Association for Computational Linguistics.

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. 2021. Explaining answers with entailment trees. *arXiv preprint arXiv:2104.08661*.

Bhavana Dalvi, Niket Tandon, Antoine Bosselut, Wen-tau Yih, and Peter Clark. 2019. Everything happens for a reason: Discovering the purpose of actions in procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4496–4505, Hong Kong, China. Association for Computational Linguistics.

Sahith Dambekodi, Spencer Frazier, Prithviraj Ammanabrolu, and Mark O Riedl. 2020. Playing text-based games with common sense. *arXiv preprint arXiv:2012.02757*.

Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, Luca Weihs, Mark Yatskar, and Ali Farhadi. 2020. RoboTHOR: An Open Simulation-to-Real Embodied AI Platform. In *CVPR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. 2019. Go-explore: a new approach for hard-exploration problems. *arXiv preprint arXiv:1901.10995*.

Angela Fan, Jack Urbanek, Pratik Ringshia, Emily Dinan, Emma Qian, Siddharth Karamcheti, Shrimai Prabhumoye, Douwe Kiela, Tim Rocktaschel, Arthur Szlam, et al. 2020. Generating interactive worlds with text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1693–1700.

Richard E Fikes and Nils J Nilsson. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3-4):189–208.

Nancy Fulda, Daniel Ricks, Ben Murdoch, and David Wingate. 2017a. What can you do with a rock? affordance extraction via word embeddings. *arXiv preprint arXiv:1703.03429*.

Nancy Fulda, Nathan Tibbetts, Zachary Brown, and David Wingate. 2017b. Harvesting common-sense navigational knowledge for robotics from uncurated text corpora. In *Conference on Robot Learning*, pages 525–534. PMLR.

Michael Genesereth, Nathaniel Love, and Barney Pell. 2005. General game playing: Overview of the aaai competition. *AI magazine*, 26(2):62–62.

Michael S. Gentry. 1998. Anchorhead.

Jonathan Gray, Kavya Srinet, Yacine Jernite, Haonan Yu, Zhuoyuan Chen, Demi Guo, Siddharth Goyal, C Lawrence Zitnick, and Arthur Szlam. 2019. Craftassist: A framework for dialogue-enabled interactive agents. *arXiv preprint arXiv:1907.08584*.

Xiaoxiao Guo, Mo Yu, Yupeng Gao, Chuang Gan, Murray Campbell, and Shiyu Chang. 2020a. Interactive fiction game playing as multi-paragraph reading comprehension with reinforcement learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7755–7765.

Xiaoxiao Guo, Mo Yu, Yupeng Gao, Chuang Gan, Murray Campbell, and Shiyu Chang. 2020b. Interactive fiction game playing as multi-paragraph reading comprehension with reinforcement learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7755–7765, Online. Association for Computational Linguistics.

William H Guss, Brandon Houghton, Nicholay Topin, Phillip Wang, Cayden Codel, Manuela Veloso, and Ruslan Salakhutdinov. 2019. Minerl: A large-scale dataset of minecraft demonstrations. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*.

Stevan Harnad. 1990. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3):335–346.

Matthew Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. 2020. Interactive fiction games: A colossal adventure. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7903–7910.

Matthew Hausknecht, Ricky Loynd, Greg Yang, Adith Swaminathan, and Jason D Williams. 2019. Nail: A general interactive fiction agent. *arXiv preprint arXiv:1902.04259*.

Ji He, Jianshu Chen, Xiaodong He, Jianfeng Gao, Lihong Li, Li Deng, and Mari Ostendorf. 2016a. Deep reinforcement learning with a natural language action space. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*

(Volume 1: Long Papers), pages 1621–1630, Berlin, Germany. Association for Computational Linguistics.

Ji He, Mari Ostendorf, Xiaodong He, Jianshu Chen, Jianfeng Gao, Lihong Li, and Li Deng. 2016b. Deep reinforcement learning with a combinatorial action space for predicting popular Reddit threads. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1838–1848, Austin, Texas. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016c. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Infocom. 1989. Learning zil.

Vishal Jain, William Fedus, Hugo Larochelle, Doina Precup, and Marc G Bellemare. 2020. Algorithmic improvements for deep reinforcement learning applied to interactive fiction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4328–4336.

Peter Jansen. 2020. Visually-grounded planning without vision: Language models infer detailed plans from high-level instructions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4412–4417.

Peter Jansen, Kelly Smith, Dan Moreno, and Huitzilin Ortiz. 2021. On the challenges of evaluating compositional explanations in multi-hop inference: Relevance, completeness, and expert ratings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Minqi Jiang, Jelena Luketina, Nantas Nardelli, Pasquale Minervini, Philip HS Torr, Shimon Whiteson, and Tim Rocktäschel. 2020. Wordcraft: An environment for benchmarking commonsense agents. *arXiv preprint arXiv:2007.09185*.

Matthew Johnson, Katja Hofmann, Tim Hutton, and David Bignell. 2016. The malmo platform for artificial intelligence experimentation. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 4246–4247.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. Qasc: A dataset for question answering via sentence composition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8082–8090.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. 2017. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*.

Bartosz Kostka, Jaroslaw Kwiecieli, Jakub Kowalski, and Pawel Rychlikowski. 2017. Text-based adventures of the golovin ai agent. In *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 181–188. IEEE.

Heinrich Küttler, Nantas Nardelli, Alexander H Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. 2020. The nethack learning environment. *arXiv preprint arXiv:2006.13760*.

P David Lebling, Marc S Blank, and Timothy A Anderson. 1979. Zork: a computerized fantasy simulation game. *Computer*, 12(04):51–59.

Andrea Madotto, Mahdi Namazifar, Joost Huizinga, Piero Molino, Adrien Ecoffet, Huaixiu Zheng, Alexandros Papangelis, Dian Yu, Chandra Khatri, and Gokhan Tur. 2020. Exploration based language learning for text-based games. *arXiv preprint arXiv:2001.08868*.

Chris Martens. 2015. Ceptre: A language for modeling generative interactive systems. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 11.

Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. 1998. Pddl-the planning domain definition language.

Vincent Micheli and François Fleuret. 2021. Language models are few-shot butlers. *arXiv preprint arXiv:2104.07972*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533.

Keerthiram Murugesan, Mattia Atzeni, Pavan Kapanipathi, Pushkar Shukla, Sadhana Kumaravel, Gerald Tesauro, Kartik Talamadupula, Mrinmaya Sachan, and Murray Campbell. 2020a. Text-based rl agents with commonsense knowledge: New challenges, environments and baselines. *arXiv preprint arXiv:2010.03790*.

Keerthiram Murugesan, Mattia Atzeni, Pushkar Shukla, Mrinmaya Sachan, Pavan Kapanipathi, and Kartik Talamadupula. 2020b. Enhancing text-based reinforcement learning agents with commonsense knowledge. *arXiv preprint arXiv:2005.00811*.

Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. 2015. Language understanding for text-based games using deep reinforcement learning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1–11, Lisbon, Portugal. Association for Computational Linguistics.

Graham Nelson. 2006. Natural language, semantic analysis, and interactive fiction. *IF Theory Reader*, 141:99–104.

Graham Nelson. 2014. The z-machine standards document version 1.1.

Vivan Raaj Rajalingam and Spyridon Samothrakis. 2019. Neuroevolution strategies for word embedding adaptation in text adventure games. In *2019 IEEE Conference on Games (CoG)*, pages 1–8. IEEE.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020a. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2020b. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*.

Ronen Tamari, Fan Bai, Alan Ritter, and Gabriel Stanovsky. 2021. Process-level representation of scientific protocols with interactive annotation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2190–2202, Online. Association for Computational Linguistics.

Ronen Tamari, Chen Shani, Tom Hope, Miriam RL Petruck, Omri Abend, and Dafna Shahaf. 2020a. Language (re) modelling: Towards embodied language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6268–6281.

Ronen Tamari, Hiroyuki Shindo, Dafna Shahaf, and Yuji Matsumoto. 2019. Playing by the book: An interactive game approach for action graph extraction from text. In *Proceedings of the Workshop on Extracting Structured Knowledge from Scientific Publications*, pages 62–71.

Ronen Tamari, Gabriel Stanovsky, Dafna Shahaf, and Reut Tsarfaty. 2020b. Ecological semantics: Programming environments for situated language understanding. *arXiv preprint arXiv:2003.04567*.

Niket Tandon, Bhavana Dalvi, Joel Grus, Wen-tau Yih, Antoine Bosselut, and Peter Clark. 2018. Reasoning about actions and state changes by injecting commonsense knowledge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 57–66, Brussels, Belgium. Association for Computational Linguistics.

Bryan Temprado-Battad, José-Luis Sierra, and Antonio Sarasa-Cabezuelo. 2019. An online authoring tool for interactive fiction. In *2019 23rd International Conference Information Visualisation (IV)*, pages 339–344. IEEE.

Chen Tessler, Tom Zahavy, Deborah Cohen, Daniel J Mankowitz, and Shie Mannor. 2019. Action assembly: Sparse imitation learning for text based games with combinatorial action spaces. *arXiv preprint arXiv:1905.09700*.

Michael Thielscher. 2010. A general game description language for incomplete information games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24.

Michael Thielscher. 2017. Gdl-iii: a description language for epistemic general game playing. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 1276–1282.

Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. 2019. Learning to speak and act in a fantasy text adventure game. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 673–683.

Ledell Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. 2018a. Starspace: Embed all the things! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. 2018b. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*.

Zhengnan Xie, Sebastian Thiem, Jaycie Martin, Elizabeth Wainwright, Steven Marmorstein, and Peter

Jansen. 2020. WorldTree v2: A corpus of science-domain structured explanations and inference patterns supporting multi-hop inference. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5456–5473, Marseille, France. European Language Resources Association.

Yunqiu Xu, Meng Fang, Ling Chen, Yali Du, Joey Tianyi Zhou, and Chengqi Zhang. 2020. Deep reinforcement learning with stacked hierarchical attention for text-based games. *Advances in Neural Information Processing Systems*, 33.

Claudia Yan, Dipendra Misra, Andrew Bennnett, Aaron Walsman, Yonatan Bisk, and Yoav Artzi. 2018. Chalet: Cornell house agent learning environment. *arXiv preprint arXiv:1801.07357*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Shunyu Yao, Karthik Narasimhan, and Matthew Hausknecht. 2021. Reading and acting while blindfolded: The need for semantics in text game agents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3097–3102.

Shunyu Yao, Rohan Rao, Matthew Hausknecht, and Karthik Narasimhan. 2020. Keep CALM and explore: Language models for action generation in text-based games. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8736–8754, Online. Association for Computational Linguistics.

Xusen Yin and Jonathan May. 2019a. Comprehensible context-driven text game playing. In *2019 IEEE Conference on Games (CoG)*, pages 1–8. IEEE.

Xusen Yin and Jonathan May. 2019b. Learn how to cook a new recipe in a new house: Using map familiarization, curriculum learning, and bandit feedback to learn families of text-based adventure games. *arXiv preprint arXiv:1908.04777*.

Xingdi Yuan, Marc-Alexandre Côté, Jie Fu, Zhouhan Lin, Chris Pal, Yoshua Bengio, and Adam Trischler. 2019. Interactive language learning by question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2796–2813, Hong Kong, China. Association for Computational Linguistics.

Xingdi Yuan, Marc-Alexandre Côté, Alessandro Sordoni, Romain Laroche, Remi Tachet des Combes, Matthew Hausknecht, and Adam Trischler. 2018. Counting to explore and generalize in text-based games. *arXiv preprint arXiv:1806.11525*.

Tom Zahavy, Matan Haroush, Nadav Merlis, Daniel J Mankowitz, and Shie Mannor. 2018. Learn what not to learn: Action elimination with deep reinforcement learning. *Advances in Neural Information Processing Systems*, 31:3562–3573.

Mikuláš Zelinka, Xingdi Yuan, Marc-Alexandre Côté, Romain Laroche, and Adam Trischler. 2019. Building dynamic knowledge graphs from text-based games. *arXiv preprint arXiv:1910.09532*.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. 2021. Scaling vision transformers. *arXiv preprint arXiv:2106.04560*.

A Extended List of Simulators

Simulators provide the infrastructure to implement the environments, objects, characters, and interactions of a virtual world, typically through a combination of a scripting engine to define the behavior of objects and agents, with a rendering engine that provides a view of the world for a given agent or user. Simulators for embodied agents exist on a fidelity spectrum, from photorealistic 3D environments to worlds described exclusively with language, where a trade-off typically exists between richer rendering and richer action spaces. This fidelity spectrum (paired with example simulators) is shown in Table 3, and described briefly below. Note that many of these higher-fidelity simulators are largely out-of-scope when discussing Text Worlds, except as a means of contrast to text-only worlds, and in the limited context that these simulators make use of text.

3D Environment Simulators: 3D simulators provide the user with complex 3D environments, including near-photorealistic environments such as AI2-Thor (Kolve et al., 2017), and include physics engines that model forces, liquids, illumination, containment, and other object interactions. Because of their rendering fidelity, they offer the possibility of inexpensively training robotic models in virtual environments that can then be transferred to the real world (e.g. RoboThor, Deitke et al., 2020). Adding objects to 3D worlds can be expensive, as this requires 3D modelling expertise that teams may not have. Similarly, adding agent actions or object-object interactions through a scripting language can be expensive if those actions are outside what is easily implemented in the simulator (like creating *gasses*, or using a pencil or saw to *modify*

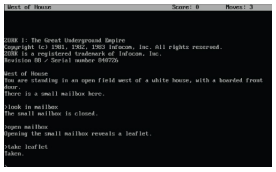
3D Environment Simulators
– AI2-Thor (Kolve et al., 2017)
– CHALET (Yan et al., 2018)
– House3D (Wu et al., 2018b)
– RoboThor (Deitke et al., 2020)
D ALFRED (Shridhar et al., 2020a)
D I ALFWorld (Shridhar et al., 2020b)
Voxel-Based Simulators
– Malmo (Johnson et al., 2016)
– MineRL (Guss et al., 2019)
Gridworld Simulators
– Rogue-in-a-box (Asperti et al., 2017)
D BABYAI (Chevalier-Boisvert et al., 2018)
I Nethack LE (Küttler et al., 2020)
I VisualHints (Carta et al., 2020)
– Griddly (Bamford, 2021)
Text-based Simulators

I Z-Machine (Infocom, 1989)
I Inform7 (Nelson, 2006)
I Ceptre (Martens, 2015)
I TextWorld (Côté et al., 2018)
I LIGHT (Urbanek et al., 2019)
I Jericho (Hausknecht et al., 2020)

Table 3: Example embodied simulation environments broken down by environment rendering fidelity. **D** specifies that environments supply natural language *directives* to the agent, **I** specifies that environments are *interacted* with (at least in part) using natural language input and/or output, and no rating represents environments that do not have a significant text component.

an object). Because of this, action spaces tend to be small, and limited to movement, and one (or a small number of) interaction commands. Some simulators and environments include **text directives** for an agent to perform, such as an agent being asked

Environment	# Actions	Examples
3D Environment Simulators		
ALFRED	7 Command	pickup, put, heat, cool
	5 Movement	move forward
Gridworld		
BABYAI	4 Command	pickup, drop, toggle
	3 Movement	turn left, move forward
NETHACK	77 Command	eat, open, kick, read
	16 Movement	move north, move east
Text-based		
ALFWorld	11 Command	goto, take, heat, clean
LIGHT	11 Command	get, drop, give, wear
	22 Emotive	applaud, wave, wink
PEG (Biomedical)	35 Command	incubate, mix, spin
Zork	56 Command	open, read, drop, drink

Table 4: Action space complexity for a selection of 3D, gridworld, and text-based environments.

to *"slice an apple then cool it"* in the ALFRED environment (Shridhar et al., 2020a). Other **hybrid environments** such as ALFWorld (Shridhar et al., 2020b) simultaneously render an environment both in 3D as well as in text, allowing agents to learn high-level task knowledge through text interactions, then ground these in environment-specific perceptual input though transfer learning.

Voxel-based Simulators: Voxel-based simulators create worlds from (typically) large 3D blocks, lowering rendering fidelity while greatly reducing the time and skill required to add new objects. Similarly, creating new agent-object or object-object interactions can be easier because they can generally be implemented in a coarser manner – though some kinds of basic spatial actions (like rotating an object in increments smaller than 90 degrees) are generally not easily implemented. Malmo (Johnson et al., 2016) and MineRL (Guss et al., 2019) offer wrappers and training data to build agents in the popular Minecraft environment. While the agent's action space is limited in Minecraft (see Table 4), the crafting nature of the game (that allows collecting, creating, destroying, or combining objects using one or more voxels) affords exploring a variety of compositional reasoning tasks with a low barrier to entry, while still using a 3D environment. Text directives, like those in CraftAssist (Gray et al., 2019), allow agents to learn to perform compositional crafting actions in this 3D environment from natural language dialog.

GridWorld Simulators: 2D gridworlds are comparatively easier to construct than 3D environments,

and as such more options are available. GridWorlds share the commonality that they exist on a discretized 2D plane, typically containing a maximum of a few dozen cells on either dimension. Cells are discrete locations that (in the simplest case) contain up to a single agent or object, while more complex simulators allow cells to contain more than one object, including containers. Agents move on the plane through simplified spatial dynamics, at a minimum *rotate left, rotate right, and move forward*, allowing the entire world to be explored through a small action space.

Where gridworlds tend to differ is in their rendering fidelity, and their non-movement action spaces. In terms of rendering, some (such as BABYAI, Chevalier-Boisvert et al., 2018) render a world graphically, using pixels, with simplified shapes for improving rendering throughput and reducing RL agent training time. Others such as NetHack (Küttler et al., 2020) are rendered purely as textual characters, owing to their original nature as early terminal-only games. Some simulators (e.g. Griddly, Bamford, 2021) support a range of rendering fidelities, from sprites (slowest) to shapes to text characters (fastest), depending on how critical rendering fidelity is for experimentation. As with 3D simulators, hybrid environments (like VisualHints, Carta et al., 2020) exist, where environments are simultaneously rendered as a Text World and accompanying GridWorld that provides an explicit spatial map.

Action spaces vary considerably in GridWorld simulators (see Table 4), owing to the different scripting environments that each affords. Some environments have a small set of hardcoded environment rules (e.g. BABYAI), while others (e.g. NetHack) offer nearly 100 agent actions, rich crafting, and complex agent-object interactions. Text can occur in the form of task directives (e.g. *"put a ball next to the blue door"* in BABYAI), partial natural language descriptions of changes in the environmental state (e.g. *"You are being attacked by an orc"* in NetHack), or as full Text World descriptions in hybrid environments.