A Spoken Dialogue System for Spatial Question Answering in A Physical Blocks World

Anonymous Author(s)
Submission Id: 1089

ABSTRACT

The blocks world is a classic toy domain that has long been used to build and test spatial reasoning systems. Despite its relative simplicity, tackling this domain in its full complexity requires the agent to exhibit a rich set of functional capabilities, ranging from vision to natural language understanding. There is currently a resurgence of interest in solving problems in such limited domains using modern techniques. In this work we tackle spatial question answering in a holistic way, using a vision system, speech input and output mediated by an animated avatar, a dialogue system that robustly interprets spatial queries, and a constraint solver that derives answers based on 3-D spatial modeling.

CCS CONCEPTS

• **Computing methodologies** → **Spatial and physical reasoning**; • **Human-centered computing** → **Interactive systems and tools**.

KEYWORDS

spatial language, spatial relations, blocks world, question answering, dialogue agent

ACM Reference Format:
Anonymous Author(s). 2019. A Spoken Dialogue System for Spatial Question Answering in A Physical Blocks World. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The past 10 or 20 years have seen rapid progress in many areas of AI technology. However, despite impressive advances in specific, narrow tasks, such as object recognition, natural language parsing and machine translation using RNNs and word and sentence embeddings, game playing, etc., there is still a shortage of multimodal interactive systems capable of performing high-level tasks requiring understanding and reasoning. Because of the complexity of most real-life tasks, the blocks world domain provides an ideal experimental setting for developing prototypes with such capabilities.

Interest in the blocks world as a domain for AI research goes back as far as the 1970s, with Winograd's thesis [14] being one of the earliest studies, utilizing a virtual environment along with text-based interaction. Recently there has been a resurgence of interest in solving problems in such limited domains using modern techniques. Despite its relative simplicity, the blocks world domain motivates implementation of diverse capabilities in a virtual interactive agent aware of physical blocks on a table, including visual scene analysis, spatial reasoning, planning, learning of new concepts, dialogue management and voice interaction, and more. In this work, we describe an end-to-end system that integrates several such components in order to perform a simple task of spatial question answering about block configurations.

2 RELATED WORK

Early studies featuring the blocks world include [14] and [3], both of which relied on a simulated environment. The latter was focused on construction planning, rather than user interaction, and as such incorporated extensive reasoning about geometric consistency and structural stability, more than descriptive aspects of the block configurations. Modern efforts in blocks world spatial language include work by Perera et al. [9, 10], which relies on spatial and structural constraints to learn user-defined structures. The work by Bisk et al. [2] focuses specifically on learning spatial relations in the blocks world using a large dataset of synthetically generated scenes. The CLEVR dataset [4] and its modified versions such as [7] inspired a flurry of projects (e.g. [8], [6] and others) on visual reasoning. While the domain and scope of CLEVR is similar to ours in some respects, e.g., in referring to colors and demonstrating counting-related reasoning, it differs significantly in others, e.g., in precluding reference to structures, while allowing reference to materials.

3 TASK DESCRIPTION

Our goal is dialogue-based question answering about spatial configurations of blocks on a table. The system is designed to answer straightforward questions such as *"Which blocks are touching some red block?", "Is the X block clear?", "Where is the Y block?",* etc. (where X and Y are unique block labels). The task of describing and answering questions about block configurations in a dialogue setting, while relatively simple, serves two purposes. First, it is function-complete in the sense that it requires most of the components needed for physical blocks world problem solving, such as dialogue management, audio-visual input-output, etc., to be in place. Once this pipeline is complete, one can proceed to add more advanced functionalities. Second, it sets the stage for our longer-term goal of building a collaborative blocks world agent, capable of interactively learning new structural concepts and building examples of them, relying on natural language communication with the user. This task requires a spatial reasoning component, operating on the ordinary kinds of spatial language used by people.

We have collected an initial sample set of about 500 question-answer pairs to evaluate the breadth of the concepts and phrasings

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

people use in describing or asking about spatial configurations. Three volunteers with no experience with the system participated in the data collection. This set was filtered to remove incoherent/contrived occurrences and the remaining filtered subset was used for guiding the development.

We roughly classify questions in one of six categories: identification, confirmation, existential, counting, descriptive, and attribute-inquiry questions. *"Which block is...?"/"What is..."* is a general template for the questions of the first category, *"Is X IN RELATION Y to Z?"* is a template for the second, etc. The support for basic spatial relations is implemented for individual blocks as arguments. We are working on the implementation for structures and regions as well, in order to be able to answer questions like *"Is the bottom block in the tallest stack red?", "Which blocks are near the front edge of the table?"*, etc.

4 BLOCKS WORLD SYSTEM OVERVIEW

Our blocks world system consists of two components: the physical apparatus and a blocks world dialogue system[1]. The physical apparatus (see Fig. 1) is comprised of a square table surface, approximately 1.5m x 1.5m in size, several cubical blocks with 0.15m sides, two Microsoft Kinect sensors mounted at the back end of the table to track the state of the world, and a display for user interaction. The blocks are marked with corporate logos, such as McDonald's, Toyota, Texaco, etc., which serve as block names and allow the user and the system to uniquely identify and refer to individual blocks. The blocks are also color-coded as either red, green, or blue, using the colored stripes running along the edges of the blocks (see the Figure).

Figure 1: The blocks world apparatus setup.

The architecture of the software component is shown in Fig. 2. The system uses audio-visual input and output. The block detection and tracking module periodically reads the input from the Kinect cameras and updates the block positioning information. Based on the information from the block tracking module, the physical block arrangement is modeled as a 3D scene in Blender. All the spatial processing is performed on that model. The automatic speech recognition (ASR) module, based on the Google Cloud Speech-To-Text API, is responsible for generating the transcripts of user utterances. For communicating back to user, we employ the interactive avatar character, David, developed by SitePal. The avatar is capable of vocalizing the text and displaying facial expressions, making the flow of conversation more natural than with textual I/O. The spatial component module together with the constraint solver is responsible for analyzing the block configuration with respect to the conditions implicit in the user's utterance. The Eta dialogue manager is responsible for *unscoped logical form* (ULF) generation (see subsections 4.1 and 4.5) and controlling the dialogue flow and transition between phases, such as greeting, ending the session, etc. Finally, the blocks world manager is the unifying component, controlling the rest of the system and facilitating the message passing and synchronization between the modules.

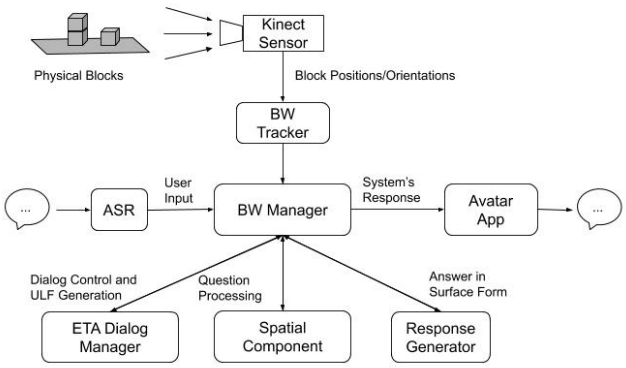

Figure 2: The blocks world dialogue pipeline. The arrows indicate the direction of interaction between the modules.

A typical round of interaction starts with the user asking a spatial question. The blocks world manager obtains and preprocesses the transcript of user's speech, and then sends it to Eta. Eta generates the ULF representation of the question and sends it back to the blocks world manager. Having received the ULF string, the blocks world manager performs query processing, which includes two stages. First, the ULF string is parsed into an internal query frame, with slots corresponding to the main predicate in the question, arguments and modifiers of the predicate (e.g., negation, etc.). The advantage of this query frame is that it provides a more canonical representation of the question than the surface form or even the ULF tree, enabling easier inference. Second, the constraint solver is applied to the query frame. It resolves the main predicate and its arguments and determines which combinations of objects present in the scene satisfy the constraints contained in the question. The constraint solver returns these combinations along with the certainty for each combination as the answer. Based on this answer set, an answer to the user's question is generated in plain English. The response generator provides expanded answers for the given questions or reports failure in case the question constraints cannot be satisfied. The English answer is then sent to the web-app

[1] A site for the implementation of all but the Kinect blocks detector will be supplied.

controlling the avatar for vocalizing it to the user, whereupon the system is ready for the next round.

4.1 Unscoped Logical Form (ULF)

We rely on ULF [5] as an intermediate question representation format. ULF is similar in purpose to the *abstract meaning representation* (AMR) in semantic parsing [1]. However, ULF is close to the surface form of English, and covers a richer set of semantic phenomena than AMR, and does so in a type-consistent way. To illustrate the approach, consider the example "Which blocks are on two other blocks?". The resulting ULF will be (((Which.d (plur block.n)) ((pres be.v) (on.p (two.d (other.a (plur block.n)))))) ?). As can be seen from this example, the resulting ULF retains much of the surface structure, but uses semantic typing and adds operators to indicate plurality, tense, aspect, and other linguistic phenomena. To facilitate conversion of English into ULF, a limited semantic parser was written, based on the typical vocabulary and phrasings occurring in spatial questions (see also section 4.5).

4.2 Query Frames

The query frame is the final representation of the question with all the components assigned to the designated slots. The structure of a query frame is recursive. An example is presented below:

```
Sentence {
    Content = Predicate {
            Content = TPrep {on}
            ARG0 = Argument {
               ObjectType = block,
               ObjectId = NULL,
               Determiner = which,
               Modifiers = [plur] }
            ARG1 = Argument {
               ObjectType = block,
               ObjectId = NULL,
               Determiner = other,
               Modifiers = [plur, TNumber {two}] }
            PredModifiers = [] }}
```

The *Sentence* frame encapsulates all the components. Its top-level entries consist of *Predicate* frames and *Argument* frames, which encapsulate the spatial relations and the entities in the blocks world, respectively. The predicate frames contain the predicate itself, its arguments, and possible predicate modifiers, e.g., *directly*, *slightly*, etc. The argument frames contain the type of the argument (block, table, structure, or a region – e.g., *front of the table*). The *ObjectId* slot is reserved for potential unique identifiers such as block names, and, potentially, unique structure identifiers. The *Determiner* slot is of course reserved for determiners, and the *Modifiers* slot contains a list of possible argument modifiers, which can be block colors, numerals, etc., perhaps including other predicates.

4.3 Constraint Solver

The constraint solver takes the above query frame as an input and processes it recursively. If the sentence contains a descriptive predicate, it first resolves its arguments, by taking the set of all the entities present in the context and applying a series of filters, so as to extract just the objects satisfying the specified attributes of the arguments. Then it resolves the predicate by mapping the content of the predicate (a word) to the actual function implementing that predicate. The constraint solver applies the resolved function to all combinations of arguments and generates a list of tuples of form ((*arg0*, *arg1*), *certainty*) where *certainty* is the numerical value representing how strongly the relation holds between the arguments. For example, for the question *"Which blue blocks are on top of the Toyota block and the Burger King Block"*, the argument resolution will return the set of blue blocks as *arg0*, e.g., assuming that the *Target*, *Texaco* and the *Toyota* blocks are blue, (*Target*, 1.0), (*Texaco*, 1.0), (*Toyota*, 1.0), and a pair ((*Toyota*, *Burger King*), 1.0) as *arg1*. In this example, since the color and identity of each block is known, the certainties are 1.0. Then the constraint solver will apply the predicate *on* to each combination, obtaining a list like ((*Target*, (*Toyota*, *Burger King*)), 0.87), ((*Texaco*, (*Toyota*, *Burger King*)), 0.76), ((*Toyota*, (*Toyota*, *Burger King*)), 0.0), where from the certainty values we can infer that the *Target* block is on top of both, the *Toyota* and the *Burger King* block, *Texaco* block is somewhat on top, and the *Toyota* block is definitely not.

4.4 Spatial Relations

The spatial relations implementation borrows from [11]. Our work extends the vocabulary of spatial relations to include orientations and structure-related locative expressions, e.g., *"Which block is the Toyota block facing?"* and *"the middle block in a row"*.

The factors that contribute to the semantics of spatial prepositions can be divided into geometric and non-geometric (functional). Geometric factors are relatively straightforward; they include locations, sizes and distances. Non-geometric factors include background knowledge about the relata—their physical properties, roles, the way we interact with them—as well as the perceived "frame" (such as a table top) and the presence and characteristics of other objects within that frame. Because of the simplicity and uniformity of objects and structures in the blocks world, geometric factors are the most important for our system. Our approach is rule-based. Each spatial relation takes one or more arguments and applies a sequence of primitive metrics evaluating various criteria, such as the distance between the object centers, whether the objects are in contact, whether they possess certain properties, etc. Each metric returns a real number from $[0, 1]$. These metrics represent contributing factors to a relation and they are combined, or the maximum among them is taken. Whenever possible we rely on approximations to the real 3D meshes of objects, using centroids and bounding boxes (smallest rectangular regions encompassing the objects). There are two main reasons for that. First, we are trying to achieve real-time performance. Second, in many circumstances, given the object shapes and distances between them, the approximations yield acceptable results. Among the basic geometric primitives used in our models are various relativized distances. An example is the distance between object centroids divided by the object sizes; that quantity will be 1.0 for cubes or spheres touching one another. This is a useful measure if the objects are convex or located relatively far apart. We introduce separate distance metrics for flat (roughly planar) or elongated (roughly linear) objects such as the table, or stacks and rows respectively.

The perceived frame, and the scale and statistics of objects in the vicinity of objects being related, are additional important factors. For some prepositions we first compute the raw value (between 0 and 1) representing the context-independent value of the preposition's metric. This metric is then modified by scaling it up or down depending on the values of the same metric for other objects in the scene. For example, suppose that the raw nearness metric $near\_raw(A, B)$ for two objects A and B is 0.55 out of 1.0. This reflects the fact that without further context, this is an ambiguous situation. However, if B is the closest object to A, i.e., $near\_raw(C, A) < 0.55, \forall C(C \neq B)$, we can say that B is *relatively* near A. In this case the final score $near(A, B)$ will be boosted by a small amount (depending on the distribution of the objects in the scene), making a more definite judgment possible.

4.5 Dialogue Manager

Eta is a dialogue manager (DM) designed to follow a modifiable dialogue schema, specified using a flexible and expressive schema language. This schema specifies a plan of expected dialogue actions of the user and system, subject to change as the interaction proceeds. The DM resembles the dialogue manager used by the LISSA system [12, 13], but allows for logical interpretation of queries. The dialogue actions are instantiated into events over the course of the conversation. Possible actions include "primitive" explicitly-defined utterances by the system, as well as abstract actions to interpret the user's response, or to form a reaction (making a reply, or initiating a subplan) to this interpretation. Using hierarchical pattern transduction methods, these abstract actions are converted into one or more primitive actions during the execution of the dialogue plan. The pattern transduction process is implemented using transduction trees, whose internal nodes specify patterns to be matched against the expression to be transduced. When all patterns at the internal nodes of a root-to-leaf path have been matched, a template associated with the final node (leaf) is filled in using match results from the parent. If a match at an internal node fails, its sibling nodes are tried, and if all siblings fail, the search continues recursively at the siblings of the parent. The result of a tree transduction, i.e., a filled-in template, is used in whatever way is specified by a directive associated with the template (send the result to output, continue in some subtree, initiate a subschema, etc.).

The system uses hierarchical pattern transductions to interpret the user input in two stages. First, the DM extracts a simple, context-independent *gist clause* by "tidying up" the user's response in the context of Eta's previous utterance. Second, if the system recognizes the gist clause as a spatial question, the gist clause is transduced into its corresponding ULF representation. This latter transduction amounts to a semantic parse using transduction trees geared towards various phrase types (NPs, PPs VPs, etc.); templates at leaf nodes in this case specify assembly of the ULFs of subphrases, obtained recursively, into a complete ULF for the targeted type of phrase. The resulting question ULF is then output to the spatial question-answering system. To generate nontechnical verbal reactions to the user, the system uses pattern transduction to construct an output from the gist clause extracted in the previous step. In the case where the gist clause represents a spatial question, the spatial question-answering system supplies the response.

4.6 Response Generation and Communicating Back to User

The response generation for spatial questions directly maps outputs from the constraint solver – in the form <arg0, relation, arg1, certainty> – to natural English answers, phrased differently for each type of spatial question that may be posed. Each question class is handled distinctly, altering the English phrasing of the constraint solver's answer so that it sounds natural and informative. However, more than just the question type is accounted for in the response. For example, most questions reveal the questioner's expectations about the desired number of answers through use of singular or plural terms, e.g., *"which block is..." vs. "which blocks are..."*. If the system finds only one object satisfying the constraints when the phrasing of the question presupposes multiple ones (such as in the latter case), the response includes corrective phrasing, such as *"Only the Toyota block..."*. Furthermore, responses typically reflect the system's degree of certainty, and in cases where the system gives both certain and uncertain answers, the response separates them and distinguishes the confident answers from the uncertain ones. When the surface form of the answer is ready, it is sent to the avatar app to vocalize it for the user.

5 PRELIMINARY EVALUATION

Since the work is still ongoing, our experimental assessment of the system remains informal. We have found that the system is capable of correctly answering the great majority (around 90%) of spatial questions in the development set, and around 70% of miscellaneous "off the cuff" spoken questions asked by multiple experimenters during live dialogue test runs. Correctness is tracked both in terms of the ULFs produced (and displayed under the "David" avatar) and in terms of the generated spoken answers. The spatial component also displays satisfactory sensitivity in terms of the certainty cut-off threshold. That is, the threshold determining which objects are included seems in accord with human intuitions.

6 CONCLUSION

We have built an end-to-end, spatial question answering system for a physical blocks world, already able to handle a majority of questions in dialogue mode. Our spatial language model relies on intuitive computational models of spatial prepositions that try to mirror human judgment by combining geometrical information with context-specific information about the objects and the scene. This enables natural interaction between the machine and the user. The ongoing work and near-term work is targeting reasoning about structures and complex shapes, which will eventually be incorporated into blocks world structure learning and collaborative construction tasks.

REFERENCES

[1] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*. 178–186.

[2] Yonatan Bisk, Kevin J Shih, Yejin Choi, and Daniel Marcu. 2018. Learning interpretable spatial operations in a rich 3d blocks world. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[3] Scott Elliott Fahlman. 1974. A planning system for robot construction tasks. *Artificial intelligence* 5, 1 (1974), 1–49.

[4] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2901–2910.

[5] Gene Louis Kim and Lenhart Schubert. 2019. A Type-coherent, Expressive Representation as an Initial Step to Language Understanding. In *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*. Association for Computational Linguistics, Gothenburg, Sweden, 13–30. https://www.aclweb.org/anthology/W19-0402

[6] Satwik Kottur, José MF Moura, Devi Parikh, Dhruv Batra, and Marcus Rohrbach. 2019. CLEVR-Dialog: A Diagnostic Dataset for Multi-Round Reasoning in Visual Dialog. *arXiv preprint arXiv:1903.03166* (2019).

[7] Runtao Liu, Chenxi Liu, Yutong Bai, and Alan L Yuille. 2019. Clevr-ref+: Diagnosing visual reasoning with referring expressions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4185–4194.

[8] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. 2019. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. *arXiv preprint arXiv:1904.12584* (2019).

[9] Ian Perera, James Allen, Choh Man Teng, and Lucian Galescu. 2018. Building and Learning Structures in a Situated Blocks World Through Deep Language Understanding. In *Proceedings of the First International Workshop on Spatial Language Understanding*. 12–20.

[10] Ian Perera, James Allen, Choh Man Teng, and Lucian Galescu. 2018. A Situated Dialogue System for Learning Structural Concepts in Blocks World. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*. 89–98.

[11] Georgiy Platonov and Lenhart Schubert. 2018. Computational Models for Spatial Prepositions. In *Proceedings of the First International Workshop on Spatial Language Understanding*. 21–30.

[12] S.Z. Razavi, M.R. Ali, T.H. Smith, L.K. Schubert, and M.E. Hoque. 2016. The LISSA virtual human and ASD teens: An overview of initial experiments. In *Proc. of the 16th Int. Conf. on Intelligent Virtual Agents (IVA 2016)*. Los Angeles, CA, 460–463.

[13] S.Z. Razavi, L.K. Schubert, M.R. Ali, and H.E. Hoque. 2017. Managing casual spoken dialogue using flexible schemas, pattern transduction trees, and gist clauses. In *5th Ann. Conf. on Advances in Cognitive Systems (ACS 2017)*. Rensselaer Polytechnic Institute, Troy, NY.

[14] Terry Winograd. 1972. Understanding natural language. *Cognitive psychology* 3, 1 (1972), 1–191.