Automatic Generation of Situation Models for Plan Recognition Problems

Kristina Y. Yordanova
University of Rostock
18059 Rostock
Germany
kristina.yordanova@uni-rostock.de

Abstract

Recent attempts at behaviour understanding through language grounding have shown that it is possible to automatically generate models for planning problems from textual instructions. One drawback of these approaches is that they either do not make use of the semantic structure behind the model elements identified in the text, or they manually incorporate a collection of concepts with semantic relationships between them. We call this collection of knowledge situation model. The situation model introduces additional context information to the model. It could also potentially reduce the complexity of the planning problem compared to models that do not use situation models. To address this problem, we propose an approach that automatically generates the situation model from textual instructions. The approach is able to identify various hierarchical, spatial, directional, and causal relations. We use the situation model to automatically generate planning problems in a PDDL notation and we show that the situation model reduces the complexity of the PDDL model in terms of number of operators and branching factor compared to planning models that do not make use of situation models.

1 Introduction

Libraries of plans combined with observations are often used for behaviour understanding (Ramirez and Geffner, 2011; Krüger et al., 2014; Yordanova and Kirste, 2015). Such approaches rely on PDDL-like notations to generate a library of plans and then reason about the agent's actions, plans, and goals based on observations. Models describing plan recognition problems for behaviour understanding are typically manually developed (Ramírez and Geffner, 2009; Ramirez and Geffner, 2011; Baker et al., 2009). The manual modelling is however time consuming and error prone and often requires domain expertise (Nguyen et al., 2013).

To reduce the need of domain experts and to reduce the time required for building the model, one can substitute them with textual data (Philipose et al., 2004). More precisely, one can utilise the knowledge encoded in textual instructions to learn the model structure. Textual instructions specify tasks for achieving a given goal without explicitly stating all the required steps. On the one hand, this makes them a challenging source for learning a model (Branavan et al., 2010). On the other hand, they are usually written in imperative form, have a simple sentence structure, and are highly organised. Compared to rich texts, this makes them a better source for identifying the sequence of actions needed for reaching the goal (Zhang et al., 2012).

According to (Branavan et al., 2012), to learn a model for planning problems from textual instructions, the system has to: 1. **extract the actions' semantics** from the text, 2. **learn the model semantics** through language grounding, 3. and finally to **translate it into computational model** for planning problems.

In this work we add the **learning of a situation model** as a requirement for learning the model structure. As the name suggests, it provides context information about the situation (Ye et al., 2012). It is a collection of concepts with semantic relations between them. In that sense, the situation model plays the role of the common knowledge base shared between different entities.

In this work, we show that a computational

Proceedings of Recent Advances in Natural Language Processing, pages 823–830,
Varna, Bulgaria, Sep 4–6 2017.
https://doi.org/10.26615/978-954-452-049-6_105

model for plan recognition problems can benefit from a situation model, which describes the semantic structure of the model elements, as it (1) introduces additional context to the model and (2) it can be used to reduce the model complexity through action specialisation. We propose a method for learning the situation model from textual instructions that relies on language taxonomies, word dependencies and implicit causal relations to identify the semantic structure of the model. We use the situation model to generate planning operators for a planning problem in a Planning Domain Definition Language (PDDL) notation. We evaluate our approach by generating a model that describes the preparation of brownies. We compare the model complexity with and without the usage of the situation model in terms of number of operators and mean branching factor.

2 Related Work

The goal of grounded language acquisition is to learn linguistic analysis from a situated context (Branavan et al., 2011; Vogel and Jurafsky, 2010). This could be done in different ways: through grammatical patterns that are used to map the sentence to a machine understandable model of the sentence (Li et al., 2010; Zhang et al., 2012; Branavan et al., 2012); through machine learning techniques (Sil and Yates, 2011; Chen and Mooney, 2011; Benotti et al., 2014; Goldwasser and Roth, 2014; Kollar et al., 2014); or through reinforcement learning approaches that learn language by interacting with an external environment (Branavan et al., 2012, 2011, 2010; Vogel and Jurafsky, 2010; Babeş-Vroman et al., 2012; Goldwasser and Roth, 2014; Kollar et al., 2014).

Models learned through language grounding have been used for plan generation (Li et al., 2010; Branavan et al., 2012), for learning the optimal sequence of instruction execution (Branavan et al., 2011, 2010), for learning navigational directions (Vogel and Jurafsky, 2010; Chen and Mooney, 2011), and for interpreting human instructions for robots to follow them (Kollar et al., 2014; Tenorth et al., 2010).

All of the above approaches have two drawbacks. The first problem is the way in which the preconditions and effects for the planning operators are identified. They are learned through explicit causal relations, that are grammatically expressed in the text (Li et al., 2010; Sil and Yates, 2011). The existing approaches, however, either rely on initial manual definition to learn these relations (Branavan et al., 2012), or on grammatical patterns and rich texts with complex sentence structure (Li et al., 2010). Textual instructions however usually have a simple sentence structure where grammatical patterns are rarely discovered (Yordanova, 2015). The existing approaches do not address the problem of discovering causal relations between sentences, but assume that all causal relations are expressed within the sentence (Tenorth et al., 2010). In textual instructions however, the elements representing cause and effect are usually found in different sentences (Yordanova, 2015).

The second problem is that existing approaches either rely on manually defined situation model (Sil and Yates, 2011; Branavan et al., 2012; Goldwasser and Roth, 2014), or do not use one (Li et al., 2010; Branavan et al., 2011, 2010; Zhang et al., 2012; Vogel and Jurafsky, 2010). However, one needs a situation model to deal with model generalisation problems and as a means for expressing the semantic relations between model elements (Yordanova and Kirste, 2016; Yordanova, 2016). What is more, the manual definition is time consuming and often requires domain experts.

To address these two problems, in previous works we outlined an approach for automatic generation of behaviour models from texts (Yordanova and Kirste, 2016; Yordanova, 2016, 2017). In this work, we extend the approach by proposing a method for automatic generation of situation models. The method adapts the idea proposed by (Yordanova, 2015) to use time series analysis to identify the causal relations between text elements. Our approach uses this idea to discover causal relations between actions. It also makes use of existing language taxonomies and word dependencies to identify hierarchical, spatial and directional relations, as well as relations identifying the means through which an action is accomplished. The situation model is then used to generate planning operators that use the situation model's semantic structure in order to specialise the operators and thus to reduce the model complexity. In the following, we describe the approach in details.

3 Approach

The goal of the work is to build a situation model for a given planning problem. In this sense, a sit-

uation model is the knowledge base containing all relevant information about a given situation. This information is represented in terms of entities describing the relevant elements for a given situation and the semantic relations between these entities. In the following we first discuss which elements are of interest for us and then we describe our approach for generating the situation model from textual instructions.

3.1 Identifying Elements of Interest

The first step in generating the situation model is to identify the elements of interest in the text. We consider a text to be a sequence of sentences divided by a sentence separator.

Each sentence in the text is then represented by a sequence of words, where each word has a tag describing its part of speech (POS) meaning.

In a text we have different types of words. We are most interested in verbs as they describe the actions that can be executed in the environment. The actions are then verbs in their infinitive form or in present tense, as textual instructions are usually described in imperative form with a missing agent.

We are also interested in those nouns that are the direct (accusative) objects of the verb. These nouns give us the elements of the world with which the agent is interacting (in other words, objects on which the action is executed).

Apart from the direct objects, we are also interested in any indirect objects of the action. Namely, any nouns that are connected to the action through a preposition. These nouns give us spacial, locational or directional information about the action being executed, or the means through which the action is executed (e.g. an action is executed "with" the help of an object). We denote the set of direct and indirect objects with O.

3.2 Building the Initial Situation Model

Given the set of objects O, the goal is to build the initial structure of the situation model from these elements. This structure consists of words, describing the elements of a situation and the relations between these elements. If we think of the words as nodes and the relations as edges, we can then represent the situation model as a graph.

Definition 1 ***(Situation model)*** *Situation model $G := (W, R)$ is a graph consisting of nodes represented through words W and of edges represented through relations R, where for two words $a, b \in W$, there exists a relation $r \in R$ such that $r(a, b)$.*

The initial structure of the situation model is represented by a taxonomy that contains the objects O and their abstracted meaning on different levels of abstraction. To do that, a language taxonomy L containing hyperonymy relations between the words of the language is used (this is the is-a relation between words).

To build the initial situation model, we start with the set O as the leaves of the taxonomy and for each object $o \in O$ we recursively search for its hypernyms. This results in a hierarchy where the bottommost layer consist of the elements in O and the uppermost layer contains the most abstract word, that is the least common parent of all $o \in O$. In that sense, a word at a higher abstraction level is the least common parent of some words on a lower abstraction level.

3.3 Extending the Situation Model

As the initial situation model contains only the abstraction hierarchy of the identified objects, we extend it by first including the list of all actions to the situation model and then adding the relations between actions and indirect objects and actions and direct objects to the graph.

On one hand, this step is performed in order to enrich the semantic structure of the model. On the other hand, it gives the basis for the planning operators, as the list of arguments in an operator is represented by all objects that are related to the action.

3.4 Adding Causal Relations

The last step is extending the situation model with causal relations. The causal relations provide the cause-effect relation between actions in the model. It is important for the planning problem, as the planning operators are defined through preconditions and effects, which in turn build up the causal structure of the planning problem.

To discover causal relations between actions in the text, we consider two cases: (1) relations between two actions in the text; (2) relations between two action-object pairs in the text. We consider the first case as there are actions that are not related to a specific direct or indirect object but that still are causally related to other actions. We consider the second case because applying one action on an

object can cause the execution of another action on the same object. We can think of the second case as a special case of the first, where we have filtered out any elements that could cause "noise" when searching for causality.

To discover causal relations between actions, we adapt the algorithm proposed by (Yordanova, 2015), which makes use of time series analysis. We start by representing each unique action (or each action-object tuple) in a text as a time series. Each element in the series is a tuple consisting of the number of the sentence in the text, and the number of occurrences of the action (tuple) in the sentence.

In order to discover causal relations based on the generated time series, we make use of the Granger causality test. It is a statistical test for determining whether one time series is useful for forecasting another. More precisely, Granger testing performs statistical significance test for one time series, "causing" the other time series with different time lags using auto-regression (Granger, 1969).

Generally, for two time series, we perform Granger test, and if the p value of the result is under the significance threshold, we conclude that the first time series causes the second, hence the first word causes the second. For example, we generate time series for the words "take" and "put" and after applying the Granger test, it concludes that the lagged time series for "take" significantly improve the forecast of the "put" time series, thus we conclude that "take" causes "put".

Now that we have identified the causal relations between actions, we add them in the situation model. We do that by adding the set of new relations to the existing set of relations in the situation model. An example of a situation model can be seen in Figure 2.

3.5 Generating Planning Operators

In order to test whether the situation model reduces the complexity of the planning problem, we generate operators based on the situation model. Figure 1 shows an example of an operator in the Planning Domain Definition Language (PDDL). It consists of action name, parameters (or arguments), and preconditions, which tell us what constraints have to be satisfied for an action to be executable, and effects, which define how the action execution changes the world.

```
(:action put
  :parameters (?o - object ?to -
      location)
  :precondition (and
          (not (executed-put ?o ?to)))
  :effect    (and
          (executed-put ?o ?to))
)
```

Figure 1: Example of an action template *put* in the PDDL notation.

To generate an operator, we take the name from the set of actions in the situation model. Then, for each action a, we take the set of arguments from the objects $o \in O$ in the situation model that have object-verb relations to the action.

The set of preconditions of an operator is then generated from the set of causal relations to actions, which cause a given action a to become executable. The set of effects consists of marking the action as executed with the given set of arguments and of negating the execution of another action if they are cyclic. Cyclic actions are actions that negate each other's effects. For example, the execution of "put the apple on the table" negates the effect of the action "take the apple". In that respect, for two operators a and b with cyclic relation, we have to negate the effects of a after executing b and vice versa, otherwise it will not be possible to execute these actions again.

4 Evaluation

To evaluate the approach, we generated a planning model from experiment instructions describing the preparation of brownies. Table 1 shows a small excerpt of the instructions.

```
1 Open the brownie bag.
2 Put the scissors in the drawer.
3 Take the brownie bag and rip the brownie bag.
4 Put the ripped brownie bag in the sink.
5 Close the drawer.
```

Table 1: Excerpt from instruction describing how to prepare brownies.

The instructions consisted of 110 short sentences describing the step by step execution of the experiment. To obtain the part of speech tags and dependencies between words, we used the Stanford NLP parser. We used the taxonomy of English language WordNet (Miller, 1995) to obtain the hyperonyms of the identified objects. As some

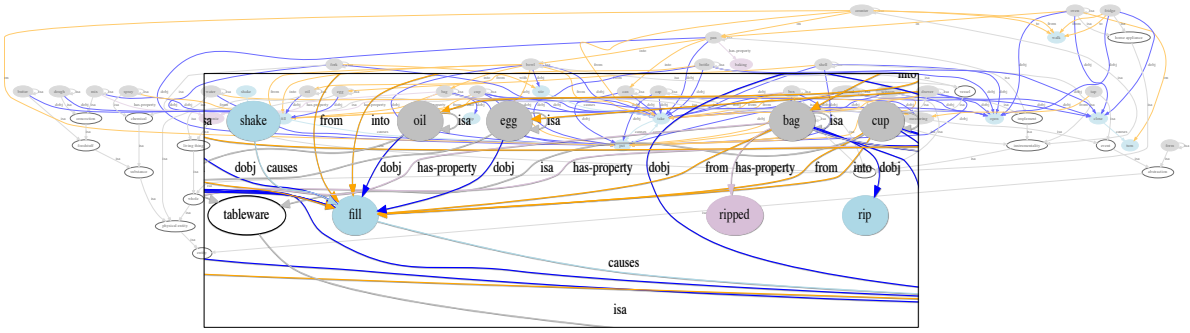

Figure 2: Extract of the situation model for the brownies instruction. Blue circles indicate actions, grey – objects, lila – properties, white taxonomy of objects. Dark blue relations indicate direct object – verb relation, yellow – different types of relations between indirect objects and verbs or nouns, light blue – causal relations, grey – abstraction hirarchy.

words have different meanings, we took the most frequently used meaning for each object. The Granger causality test was implemented in R in order to discover causal relations between the actions. Finally, the automatic generation of the situation model and the PDDL model were implemented in Haskell.

We generated a situation model which consisted of 29 objects and 10 actions identified in the text. Furthermore, 38 unique hyperonyms were identified for the 29 objects as well as 6 hyperonyms based on the relations of the indirect objects to the action. Finally, 12 causal relations were discovered. The resulting situation model contained 138 unique relations between the above identified elements. Figure 2 shows the resulting situation model for the brownies.

To evaluate the situation model, we investigated two hypotheses.

H1 The situation model provides additional context knowledge and semantic structure to the planning problem.

H2 The situation model reduces the planning model complexity compared to models that do not use situation models or only use manually defined situation models.

To investigate H1, we compared the PDDL model that makes use of the situation model ($PDDL_{sm}$) to: 1. a PDDL model containing only actions and no arguments. We call this model $PDDL_1$; 2. a PDDL model containing only the unique actions – arguments pairs discovered in the instructions. We call this model $PDDL_2$.

To investigate H2, we compared $PDDL_{sm}$ to: 1. a PDDL model that does not use a situation model. That is, each action template in the model has the same number of parameters, but they are all of the same type. In other words, any object can be used as argument for this action. We call this model $PDDL_3$; 2. a PDDL model that makes use of the hyperonyms extracted through WordNet. We assume this model represents a manually built situation model. We call this model $PDDL_4$; 3. a PDDL model that makes use of the hyperonyms extracted through WordNet and of the abstraction achieved through the relations between indirect objects and actions. We call this model $PDDL_5$.

We use the following metrics: number of operators and mean branching factor.

4.1 Results

The results for H1 can be seen in Table 2. They show that $PDDL_{sm}$ has much more operators than $PDDL_1$ and $PDDL_2$. This is apparent, as $PDDL_1$ uses only the action classes and does not have any arguments. $PDDL_2$ has arguments but these are only the concrete action – arguments pairs discovered in each sentence, so the model does not make any other combinations of arguments that might be applicable to the same action. $PDDL_{sm}$, however, generates the operators based on the identified PDDL action templates, making use of the situation model. This allows for various combinations of arguments when grounding the templates. On the one hand, this has the positive effect of actions and plan variability. The model will be able to explain many more varia-

Metrics	$PDDL_{sm}$	$PDDL_1$	$PDDL_2$
N: operators	1426	11	74
mean br. factor	1071	11	74

Table 2: Comparison between $PDDL_{sm}$, $PDDL_1$, and $PDDL_2$.

tions in the behaviour of the agent then $PDDL_1$ and $PDDL_2$. On the other hand, the high number of operators also increases the mean branching factor of the model. In other words, it would be much easier to recognise the correct actions and plan of the agent in the case of 11 (respectively 74) choices than 1136.

The results for H2 show that the situation model generated through our approach reduces the complexity of the model compared to models, which do not make use of situation models or use manually defined situation models (see Table 3). Table 3 shows that $PDDL_3$ has the highest number of operators (3053), which is to be expected as each action template can be grounded with any of the available objects. The number of operators in $PDDL_4$ decreases compared to $PDDL_3$ (1736). This is due to the introduced type hierarchy extracted through WordNet. The number of operators decreases further when one takes into account the additional relations identified between indirect objects and actions ($PDDL_5$ with 1426 operators). Adding the causal relations does not decrease the number of operators ($PDDL_{sm}$) compared to $PDDL_5$. It is also to be expected as the causal relations are not part of the type hierarchy.

Metrics	$PDDL_{sm}$	$PDDL_3$	$PDDL_4$	$PDDL_5$
oper.	1426	3053	1736	1426
br. fac.	1071	3053	1736	1426

Table 3: Comparison between $PDDL_{sm}$, $PDDL_3$, $PDDL_4$, and $PDDL_5$.

The causal relations, however, reduce the mean branching factor of the model. It decreases from 1426 in $PDDL_5$ to 1071 in $PDDL_{sm}$. In other words, in the rest of the models each action is always executable as there are no constraints to reduce the number of applicable actions. Adding the causal relations to operators reduces the branching factor allowing for better execution of the correct action. This is also visible in Figure 3. It shows the branching factor over time given a plan the model had to explain. The plan consists of 66 actions and was manually built by watching the video log of the brownies experiment. It can be seen that for the models, which do not make use of causal relations, the branching factor is constant. For $PDDL_{sm}$, however, we see both fluctuation and reduction of the branching factor, which is due to the preconditions and effects introduced in the model.

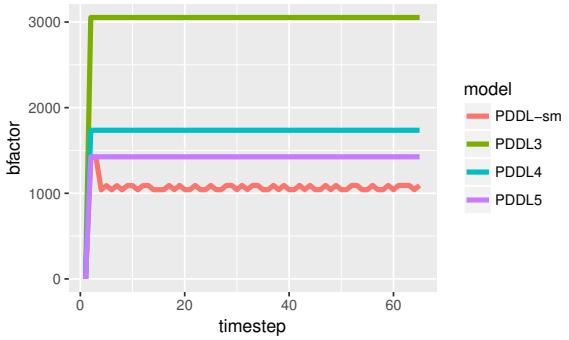

Figure 3: Branching factor for the different models for each time step.

5 Discussion and Conclusion

In this work we proposed an approach that generates situation models from textual instructions. It then uses the situation model to generate PDDL operators for planning problems. The results showed that the situation model introduces additional context information and semantic structure to the PDDL model. It also reduces the model complexity compared to models, which do not use situation models or which use manually developed situation models.

The automatically generated situation model provides valuable additional context information about the semantic structure behind the executed behaviour. It can potentially be used as a means to reason beyond the actions and the goal in the executed plan, namely by providing information about the situation in which the agent is acting.

Acknowledgments

This work is funded by the German Research Foundation (DFG) within the context of the project TextToHBM, grant number YO 226/1-1.

References

Monica Babeş-Vroman, James MacGlashan, Ruoyuan Gao, Kevin Winner, Richard Adjogah, Marie desJardins, Michael Littman, and Smaranda Muresan. 2012. Learning to interpret natural language

instructions. In *Proceedings of the Second Workshop on Semantic Interpretation in an Actionable Context*. Association for Computational Linguistics, Stroudsburg, PA, USA, SIAC '12, pages 1–6. http://dl.acm.org/citation.cfm?id=2390927.2390928.

Chris L. Baker, Rebecca Saxe, and Joshua B. Tenenbaum. 2009. Action understanding as inverse planning. *Cognition* 113(3):329–349.

Luciana Benotti, Tessa Lau, and Martín Villalba. 2014. Interpreting natural language instructions using language, vision, and behavior. *ACM Trans. Interact. Intell. Syst.* 4(3):13:1–13:22. https://doi.org/10.1145/2629632.

S. R. K. Branavan, Nate Kushman, Tao Lei, and Regina Barzilay. 2012. Learning high-level planning from text. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '12, pages 126–135. http://dl.acm.org/citation.cfm?id=2390524.2390543.

S. R. K. Branavan, David Silver, and Regina Barzilay. 2011. Learning to win by reading manuals in a monte-carlo framework. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*. Association for Computational Linguistics, Stroudsburg, PA, USA, HLT '11, pages 268–277. http://dl.acm.org/citation.cfm?id=2002472.2002507.

S. R. K. Branavan, Luke S. Zettlemoyer, and Regina Barzilay. 2010. Reading between the lines: Learning to map high-level instructions to commands. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '10, pages 1268–1277. http://dl.acm.org/citation.cfm?id=1858681.1858810.

David L. Chen and Raymond J. Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-2011)*. pages 859–865. http://www.cs.utexas.edu/users/ai-lab/?chen:aaai11.

Dan Goldwasser and Dan Roth. 2014. Learning from natural instructions. *Machine Learning* 94(2):205–232. https://doi.org/10.1007/s10994-013-5407-y.

C. W. J. Granger. 1969. Investigating Causal Relations by Econometric Models and Cross-spectral Methods. *Econometrica* 37(3):424–438. https://doi.org/10.2307/1912791.

Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. 2014. Grounding verbs of motion in natural language commands to robots. In Oussama Khatib, Vijay Kumar, and Gaurav Sukhatme, editors, *Experimental Robotics*, Springer Berlin Heidelberg, volume 79 of *Springer Tracts in Advanced Robotics*, pages 31–47. https://doi.org/10.1007/978-3-642-28572-1_3.

Frank Krüger, Martin Nyolt, Kristina Yordanova, Albert Hein, and Thomas Kirste. 2014. Computational state space models for activity and intention recognition. a feasibility study. *PLoS ONE* 9(11):e109381. https://doi.org/10.1371/journal.pone.0109381.

Xiaochen Li, Wenji Mao, Daniel Zeng, and Fei-Yue Wang. 2010. Automatic construction of domain theory for attack planning. In *IEEE International Conference on Intelligence and Security Informatics (ISI), 2010*. pages 65–70. https://doi.org/10.1109/ISI.2010.5484775.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM* 38(11):39–41. https://doi.org/10.1145/219717.219748.

Tuan A Nguyen, Subbarao Kambhampati, and Minh Do. 2013. Synthesizing robust plans under incomplete domain models. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, Curran Associates, Inc., pages 2472–2480. http://papers.nips.cc/paper/5120-synthesizing-robust-plans-under-incomplete-domain-models.pdf.

Matthai Philipose, Kenneth P. Fishkin, Mike Perkowitz, Donald J. Patterson, Dieter Fox, Henry Kautz, and Dirk Hahnel. 2004. Inferring activities from interactions with objects. *IEEE Pervasive Computing* 3(4):50–57. https://doi.org/10.1109/MPRV.2004.7.

Miquel Ramírez and Hector Geffner. 2009. Plan recognition as planning. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, IJCAI'09, pages 1778–1783. http://dl.acm.org/citation.cfm?id=1661445.1661731.

Miquel Ramirez and Hector Geffner. 2011. Goal recognition over pomdps: Inferring the intention of a pomdp agent. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*. AAAI Press, Barcelona, Spain, volume 3 of *IJCAI'11*, pages 2009–2014. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-335.

Avirup Sil and Alexander Yates. 2011. Extracting strips representations of actions and events. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*. RANLP 2011 Organising Committee, Hissar, Bulgaria, pages 1–8. http://aclweb.org/anthology/R11-1001.

M. Tenorth, D. Nyga, and M. Beetz. 2010. Understanding and executing instructions for everyday manipulation tasks from the world wide web. In *IEEE International Conference on*

Robotics and Automation (ICRA). pages 1486–1491. https://doi.org/10.1109/ROBOT.2010.5509955.

Adam Vogel and Dan Jurafsky. 2010. Learning to follow navigational directions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '10, pages 806–814. http://dl.acm.org/citation.cfm?id=1858681.1858764.

Juan Ye, Simon Dobson, and Susan McKeever. 2012. Review: Situation identification techniques in pervasive computing: A review. *Pervasive Mob. Comput.* 8(1):36–66. https://doi.org/10.1016/j.pmcj.2011.01.004.

Kristina Yordanova. 2015. Discovering causal relations in textual instructions. In *Recent Advances in Natural Language Processing*. RANLP 2015 Organising Committee, Hissar, Bulgaria, pages 714–720. http://www.aclweb.org/anthology/R15-1091.

Kristina Yordanova. 2016. From textual instructions to sensor-based recognition of user behaviour. In *Companion Publication of the 21st International Conference on Intelligent User Interfaces*. ACM, New York, NY, USA, IUI '16 Companion, pages 67–73. https://doi.org/10.1145/2876456.2879488.

Kristina Yordanova. 2017. TextToHBM: A generalised approach to learning models of human behaviour for activity recognition from textual instructions. In *Proceedings of the AAAI Workshop on Plan, Activity and Intent Recognition (PAIR)*. AAAI, San Francosco, USA, pages 891–898. https://www.aaai.org/ocs/index.php/WS/AAAIW17/paper/view/15110.

Kristina Yordanova and Thomas Kirste. 2015. A process for systematic development of symbolic models for activity recognition. *ACM Transactions on Interactive Intelligent Systems* 5(4):20:1–20:35. https://doi.org/10.1145/2806893.

Kristina Yordanova and Thomas Kirste. 2016. Learning models of human behaviour from textual instructions. In *Proceedings of the 8th International Conference on Agents and Artificial Intelligence (ICAART 2016)*. Rome, Italy, pages 415–422. https://doi.org/10.5220/0005755604150422.

Ziqi Zhang, Philip Webster, Victoria Uren, Andrea Varga, and Fabio Ciravegna. 2012. Automatically extracting procedural knowledge from instructional texts using natural language processing. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*. European Language Resources Association (ELRA), Istanbul, Turkey, pages 520–527. ACL Anthology Identifier: L12-1094. http://www.lrec-conf.org/proceedings/lrec2012/pdf/244_Paper.pdf.