Challenges in Modelling Cooking Task Execution for User Assistance

Tomasz Sosnowski
tomasz.sosnowski@uni-rostock.de
University of Rostock
Rostock, Germany

Teodor Stoev
teodor.stoev@uni-greifswald.de
University of Greifswald
Greifswald, Germany

Thomas Kirste
thomas.kirste@uni-rostock.de
University of Rostock
Rostock, Germany

Kristina Yordanova
kristina.yordanova@uni-greifswald.de
University of Greifswald
Greifswald, Germany

ABSTRACT

Executing a complex physical task according to an instruction or a checklist is typical for various fields, such as aviation or healthcare. It is possible that the person is inexperienced or under stress and therefore unable to promptly consult the instruction text for correct execution of the task. To address this problem different works propose the usage of automated assistive systems that guide the user through the task execution. This is also addressed by the Defense Advanced Research Projects Agency (DARPA) Perceptually-enabled Task Guidance (PTG) program, aiming to provide users with augmented reality goggle capable of tracking the state of the user during task execution and to display hints relevant for the task.

In this paper we discuss one important part of this project, namely, the ability to track the state of the user and the environment in order to be able to assist them. We discuss our modelling approach for the development of a probabilistic state tracker, which uses sensor observations to identify the current state and actions of the user, as well as the goal they are pursuing. The paper provides useful insights into the modelling of user behaviour, the challenges associated with modelling multi-user behaviour and how to tackle them.

CCS CONCEPTS

• **Computing methodologies** → **Planning and scheduling**; **Probabilistic reasoning**.

KEYWORDS

guidance, user modelling, state tracking, activity recognition

ACM Reference Format:
Tomasz Sosnowski, Teodor Stoev, Thomas Kirste, and Kristina Yordanova. 2023. Challenges in Modelling Cooking Task Execution for User Assistance. In *8th international Workshop on Sensor-Based Activity Recognition and Artificial Intelligence (iWOAR 2023), September 21–22, 2023, Lübeck, Germany.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3615834.3615852

iWOAR 2023, September 21–22, 2023, Lübeck, Germany

ACM ISBN 979-8-4007-0816-9/23/09.
https://doi.org/10.1145/3615834.3615852

1 INTRODUCTION

1.1 The Perceptually-enabled Task Guidance Project

The Perceptually-enabled Task Guidance (PTG) is a Defense Advanced Research Projects Agency (DARPA) program aiming to *help users perform complex physical tasks*[1]. Within the PTG program, the AMIGOS[2] project aims to provide users with guidance in the form of instructions displayed on augmented reality (AR) goggles to help them execute a certain task and avoid errors while doing so.

Two example use cases are cooking and engine maintenance. In the case of cooking, the user is supposed to follow a cooking recipe displayed on a head-up display, and engine maintenance involves inspection of an air filter of a Honda GX240 engine.

In both applications, the system needs to correctly track the state of the world and be able to identify the step the user is currently executing or trying to execute. Based on that information, guidance is provided, involving error detection and avoidance. Part of the problem is also the transfer of knowledge from textual, natural-language instructions (i.e. cooking recipes) into a structured form that can be used by the system.

1.2 State tracking with CCBM

As a part of the AMIGOS project we develop a state tracker, which can identify the current stage in the instruction execution based on the sensor (camera) observations. This is necessary for guiding the user by displaying the instruction text relevant to the current stage of instruction, but also to detect any error or deviation. Here we present the cooking use-case, which is characterized by having cooking recipes with clear steps, several ingredients and utensils, but in many cases the steps could potentially be further decomposed into more atomic actions (see Code 1).

1.3 Contribution

In the following we present our approach to modelling the user behaviour in cooking tasks with the ultimate goal to assist the user in the completion of their task. Our contribution consists of presenting our strategies for user modelling and in discussing the challenges in modelling multi-user behaviour and the potential solutions.

[1] https://www.darpa.mil/program/perceptually-enabled-task-guidance
[2] Autonomous Multimodal Ingestion for Goal Oriented Support

2 RELATED WORK

Assisting humans in completing the tasks using Markov models have been researched by several authors. In particular, [2, 3] uses partially observable Markov decision process (POMDP) to assist persons with dementia in everyday tasks (washing hands). Similar to our project, their approach utilizes observations generated by visual sensors (cameras) without a need for visual markers. Based solely on visual input, they track the hands and objects of interest, estimating the progress of the task.

Similar approach can be used to identify the goal(s) of the user performing some actions, that are observed and tracked [1]. Goal estimation aims to discover, which of the predefined end states (goals) the user is trying to reach, based on the behavior of the user (actions taken) up to the point of estimation.

Other works propose the usage of dynamic Bayesian networks and symbolic models to track the person's actions, states and goals in a probabilistic manner [4, 7]. This work is based on the approach proposed by Krüger et al. [4].

3 PRELIMINARIES

For the purpose of state tracking, we use the CCBM framework, developed at the University of Rostock [4]. CCBM stands for Computational Causal Behavior Models and it is a probabilistic engine – more precisely, a Dynamic Bayesian Network. The model is represented graphically in Figure 1, where X denotes state, with D – duration, U – starting time, G – goal, A – action, and S – state of the environment. This state is reflected by the observation Y, consisting of V – observation of current time, Z – observation of an action, and W – observation of an environment (state of an environment). In Figure 1, nodes with single outline denote hidden random variables, nodes with double outline denote observable random variables and boxes represent tuples of variables. Edges represent dependencies of a (tuple of) random variable(s) from another (tuple of) random variable(s) [4].

CCBM tracks the state of the world and actions based on observations. Being a probabilistic engine, it is able to handle ambiguous, noisy input.

Dataflow within the larger AMIGOS system is depicted in Figure 2. Camera images are processed and objects in the field of view recognized, with the perceptual translation component sending information about them to the state tracker. The tracker first stabilizes the observations to eliminate jitter (flickering of objects in and out of existence) and then generates a new observation for CCBM at regular intervals. CCBM is able to assign probability of the recipe step being already executed by the current point in time, and this information is passed down to guide the user.

CCBM uses an internal model of the world described in PDDL-like[3] notation (see Code 1). In difference to vanilla PDDL, here a predicate `duration` was used to allow modelling of actions that take some span of time, instead of being completed instantaneously. In this example, duration of the action is approximated as a gaussian distribution with mean of 3 and standard deviation of 2. Importantly, these numbers express time not in units like seconds or minutes, but rather in timesteps, at which new observations are incoming.

[3]Planning Domain Definition Language

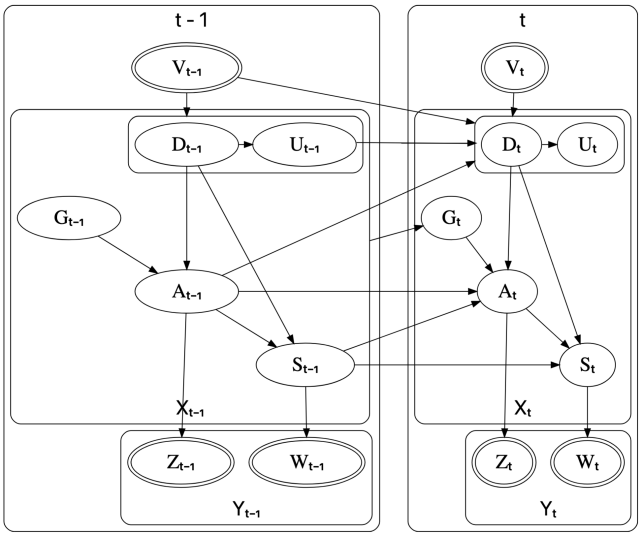

Figure 1: CCBM Dynamic Bayesian Network (Based on: [4])

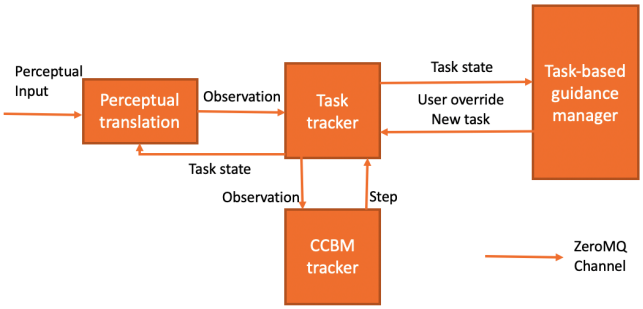

Figure 2: CCBM state tracker as a part of the system

Predicate `observation` is used to bind the PDDL code to the observation model written in *C++*, which extends the CCBM code itself and is responsible for the actual processing of the input (observations).

```
(:action transferLiquid
    :parameters ( ?v1 ?v2 - vessel ?l - liquid)
    :duration (normal 3 2)
    :precondition (is-in ?l ?v1)
    :effect (and (is-in ?l ?v2)
                 (not (is-in ?l ?v1))
                 (executed step1))
    :observation (setActivity (activity-id pour)))
```

Code 1: Sample action as defined in PDDL, with additional predicates duration and observation

In difference to the textual instruction, which specifies the preferred order of steps, users can deviate from it with various consequences – not necessarily negative ones. Some of the deviations made by the user are equally good as the original instruction, in sense that they also lead to the goal (see Figure 3). For example, when making a coffee, it does not matter whether we first insert a

paper filter into a dripper or boil the water – and it would be too restrictive to force the user to perform these actions in a particular order. What matters, however, is that both of these actions must be completed before pouring water from the kettle to the filter. In case this condition is violated, the user has committed an error that is hard to recover from (you cannot just "un-pour" the water). Such situation must be at least detected with notification of failure being send, but at best the user should be warned beforehand.

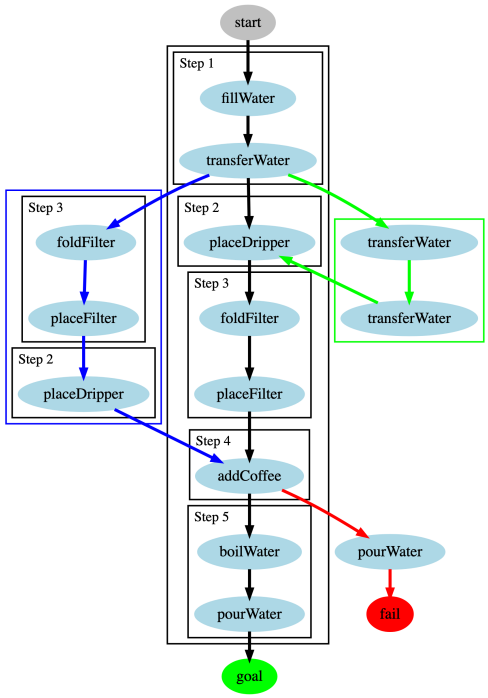

Figure 3: Possible deviations from the prescribed order of actions (sequence in black)

Fortunately, PDDL allows for a free order of action execution, as long as preconditions are satisfied. This way, we can model user's actions without a need to arrange them in an immutable sequence.

Each possible combination of the PDDL predicate values is a *state*. This corresponds to some "snapshot" of the real world, a particular arrangement of objects being in the field of vision at a given time.

In the wider AMIGOS system, CCBM is used to track tasks and is located between perceptual translation and guidance manager. Perceptual translation stack delivers new observation each time something changes in the world, then task tracker parses that input and generates observations for CCBM at regular intervals. Finally, the CCBM output is used to guide the user.

4 MODELLING APPROACH

4.1 Steps and actions

Consider the following recipe:

(1) Fill a measuring cup with water, then transfer the water to the kettle
(2) Place the dripper on a mug
(3) Fold a paper filter and place it on the dripper
(4) Put ground coffee in the filter
(5) Boil water in a kettle and pour it through the filter

This recipe consists of five *steps*, understood as the steps of an instruction as provided in natural language by the instruction text, on which the entire model is based. Each of these *steps* is reflected in PDDL domain as one or more *actions*. For example, the *action* `transferLiquid` shown in Code 1 corresponds to a fragment of step 1 in the recipe above – a complex step, that can be decomposed into separate, atomic actions of filling cup with water and transferring it to another vessel.

Of course, all *actions* in PDDL are defined as action schemas compatible with all possible parameters of a given type. Hence, we do not use an action defined specifically for transferring water from the measuring cup to the kettle, rather in Code 1 we see a use of an action schema for transferring any liquid between any two vessels, with water, cup and kettle being defined as objects that the planner needs to use in order to reach the goal.

4.2 Observations

Our state tracker accepts observations in the form of a one-dimensional vector of numbers from 0.0 to 1.0, where numbers are confidence values corresponding to the input features. Each such vector is a single observation corresponding to a single timestep.

Sample input for CCBM tracker has been shown in Table 1, where three input features describe whether the measuring cup is empty, if said cup is over the kettle, and if the dripper is near the mug. These features – a subset of all features available – are used to recognise the first two steps of the recipe above. Each row of the table is a single observation, that is, a vector of confidence values for the features in a given timestep.

Time	isEmpty(cup)	over(cup, kettle)	near(mug, dripper)
0	1.0	0.0	0.0
1	0.8	0.0	0.0
2	1.0	0.0	0.0
3	0.0	1.0	0.0
4	0.0	1.0	0.0
5	0.0	0.8	0.0
6	0.0	0.0	0.8
7	0.0	0.0	0.8

Table 1: Sample input for CCBM with three features and their changing confidence values over time

4.3 Output

For the input given in Table 1, output is provided in Table 2. For each timestep, we calculate the probability that, by the end of that timestep, a step of the instruction (recipe) has been completed. Results of that calculation have been added to the table as "steps" columns. This information is then passed down to the Guidance Manager and is used to display the status and hints for the user.

T	empty(c)	over(c,k)	near(m,d)	Step 1	Step 2
0	1.0	0.0	0.0	0.632139	0
1	0.8	0.0	0.0	0.632139	0.632181
2	1.0	0.0	0.0	0.644141	0.857809
3	0.0	1.0	0.0	0.869371	0.857809
4	0.0	1.0	0.0	0.951947	0.857809
5	0.0	0.8	0.0	0.982323	0.857809
6	0.0	0.0	0.8	0.982323	0.947729
7	0.0	0.0	0.8	0.982323	0.980772

Table 2: Sample input together with output (names of the objects abbreviated to conserve space)

5 DISCUSSION: CHALLENGES IN MULTI-TASKING

The goal of the multi-tasking is to track the execution of up to 5 cooking recipes. The actions in a multi-tasking setup are interleaved, i.e. the user can start preparing an oatmeal, but in the middle of the task boil water, go back to oatmeal and finally use boiling water to brew tea.

In addition, each utensil and ingredient can occur on a kitchen table in multiple instances.

5.1 Keeping track of an object

A single utensil can not only be used multiple times during execution of a recipe, but can be used for actions involving more than one recipe. For example, spoon can be used for preparing both tea and oatmeal.

Spoon is not necessarily a problem – it can be easily reused and we do not need to keep track of an individual spoon, rather we just need to know, if one is available. This is not the case with every utensil, however. Vessels such as measuring cups can hold some liquid or other ingredient and we need to make sure that the vessel we are operating on contains desired ingredient or is empty. Even the spoon mentioned above can become problematic if it requires cleaning between uses.

CCBM allows to model objects and their types, so we can model each instance of the vessel present on a kitchen table as a separate object. Then, actions are performed on a particular object of a known state.

As seen in Code 1, action `transferLiquid`, used for transferring liquid between two vessels, is generic and can accept any two parameters of the type `vessel`. Provided that the observations actually distinguish between different utensils, the individual identity of these utensils will be preserved – a grounded action contains a particular instance of an object and not a generic type (ie. `kettle1` instead of a generic `boilingMachine`).

5.2 Overlap between recipes in actions, utensils and ingredients

Recipes will contain some overlap between them in all elements: actions, utensils and ingredients. For example, the action of boiling the water in a kettle will be common for both coffee and tea brewing. Banana slices (and the action of slicing bananas) can be used for both oatmeal and dessert quesadillas.

Fortunately, CCBM actions and objects do not need to be bound to a particular recipe (goal). Rather, actions are generic enough to be used with any object of a given type, and conversely, objects can be operated on by any action that uses parameters of that type.

5.3 Activities taking place in a background

Not all activities are executed directly by the user. For example, water is being boiled by a kettle, with the user being free to perform other activities in the meantime.

We can handle this by decomposing the action such as "boil water" into atomic actions performed directly by the user such as "turn on the kettle" and "transfer water from kettle".

6 CONCLUSION

In this paper we presented our modelling approach for tracking the execution of cooking tasks based on instructions from recipes. We also discussed the challenges associated with modelling multi-user behaviour and how we intend to address them in our model.

In the future, we plan to extend the modelling approach by relying on models automatically generated from textual instructions as proposed in [5, 6]. This will help us to develop such models in a faster way by just adjusting them in case of inconsistencies.

ACKNOWLEDGMENTS

The AMIGOS project is funded by Defense Advanced Research Projects Agency, project number HR001122C0009. Teodor Stoev and Kristina Yordanova are partially funded by the German Research Foundation, project number YO 226/3-1.

REFERENCES

[1] Chris L. Baker, Rebecca Saxe, and Joshua B. Tenenbaum. 2009. Action understanding as inverse planning. *Cognition* 113, 3 (2009), 329–349. https://doi.org/10.1016/j.cognition.2009.07.005 Reinforcement learning and higher cognition.

[2] Jesse Hoey, Thomas Plötz, Dan Jackson, Andrew Monk, Cuong Pham, and Patrick Olivier. 2011. Rapid specification and automated generation of prompting systems to assist people with dementia. *Pervasive and Mobile Computing* 7, 3 (2011), 299–318. https://doi.org/10.1016/j.pmcj.2010.11.007 Knowledge-Driven Activity Recognition in Intelligent Environments.

[3] Jesse Hoey, Pascal Poupart, Axel von Bertoldi, Tammy Craig, Craig Boutilier, and Alex Mihailidis. 2010. Automated handwashing assistance for persons with dementia using video and a partially observable Markov decision process. *Computer Vision and Image Understanding* 114, 5 (2010), 503–519. https://doi.org/10.1016/j.cviu.2009.06.008 Special issue on Intelligent Vision Systems.

[4] 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 (11 2014), 1–24. https://doi.org/10.1371/journal.pone.0109381

[5] Teodor Stoev, Tomasz Sosnowski, and Kristina Yordanova. 2023. A Tool for Automated Generation of Domain Specific Symbolic Models From Texts. In *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops).* 276–278. https://doi.org/10.1109/PerComWorkshops56833.2023.10150252

[6] Kristina Yordanova. 2018. Extracting Planning Operators from Instructional Texts for Behaviour Interpretation. In *German Conference on Artificial Intelligence.* Berlin, Germany, 215–228. https://doi.org/10.1007/978-3-030-00111-7_19

[7] Kristina Yordanova , Stefan Lüdtke, Samuel Whitehouse, Frank Krüger, Adeline Paiement, Majid Mirmehdi, Ian Craddock, and Thomas Kirste. 2019. Analysing Cooking Behaviour in Home Settings: Towards Health Monitoring. *Sensors* 19, 3 (2019). https://doi.org/10.3390/s19030646