MuMMER

MULTIMODAL MALL ENTERTAINMENT ROBOT mummer-project.eu

Grant No. 688147. Project started 2016-03-01. Duration 48 months.

DELIVERABLE D4.1
SOCIALLY ENGAGING INTERACTION MANAGEMENT COMPONENT (INITIAL)

Christian Dondrup (HWU), Ioannis Papaioannou (HWU), Oliver Lemon (HWU), Amit Kumar (SBRE)

Beneficiaries:	**HWU** (lead), IDIAP, SBRE, LAAS, GLA
Workpackage:	**WP4: Socially appropriate and entertaining dialogue planning and action selection**

Version:	Draft 1
Nature:	OTHER (Software)
Dissemination level:	Public (PU)
Pages:	11
Date:	2017-2-27

This project has received funding from the European Union’s Horizon 2020 research and innovation programme.

Contents

1 Description

As a software deliverable, D4.1 presents two main pieces of work up to month 12 of the project: 1) the advances that have been made at HWU in developing an initial prototype system for engaging social interaction management (see Sec. 1.1), 2) work at SBRE on updating the robot knowledge-base through dialogues with humans (see Sec. 1.2). All concepts presented in the following have been implemented and can be run on the robot.

1.1 Initial MuMMER Prototype (HWU)

Monitoring the social belief state (from WP3) and the information provided by the sensors (from WP2), the HWU system selects socially appropriate engaging, and entertaining robot actions, as determined through the co-design process in WP1. To this end, we built a representation of the social belief state, based on the information provided by WP2 and WP3 which is represented as logical predicates in a database, e.g. `(looking_at_robot id_5)` representing that the person with `id_5` is looking at the robot or `(robot_dist close id_5)` representing that the robot is close to person 5. Based on this initial model for belief monitoring, a hand-crafted action selection process using a partial order forward chaining planner selects the order in which the actions should be executed. The execution itself is handled by a so-called Petri-Net Plan which transforms the output of the planning component into a specialised Finite State Machine (FSM) to achieve the given task. This allows for fast and robust execution while handling concurrency and possible recovery behaviours.

Another part of the developed prototype is the dialogue component which is used once a user is engaged in interaction with the robot. This dialogue component is currently implemented as a Markov Decision Process (MDP) trained from simulated user data in a shopping mall domain. Currently, the vast majority of language processing approaches used in robotics focus on the execution of tasks. In the MuMMER project, we aim to build a task oriented and entertaining agent which is why in the presented system the task execution is combined with a chatbot to be able to respond to utterances which do not refer to specific tasks the robot has to fulfil.

For the first year scenario, we focus on the following tasks:

- Greeting a person
- Giving directions to shops
- Giving vouchers for ongoing promotions
- Taking pictures with the robot
- Playing a quiz game
- Prompting the user for a task
- Chatting

The system shown in Figure 1 and described in Section 2 has been specifically designed to achieve all these actions. All components have been developed using the Robot Operating System (ROS) Indigo to achieve maximum impact and re-usability.

1.2 SBRE Learning System

In addition to this prototype system, SBRE developed a learning dialogue approach as described in Section 3 which has been implemented in the NAOqi framework and aims to make the robot capable of extracting relevant information while interacting with the human and enriching its knowledge base. The objective is to make the robot's dialogue selection capabilities dynamic, rich, and proactive.

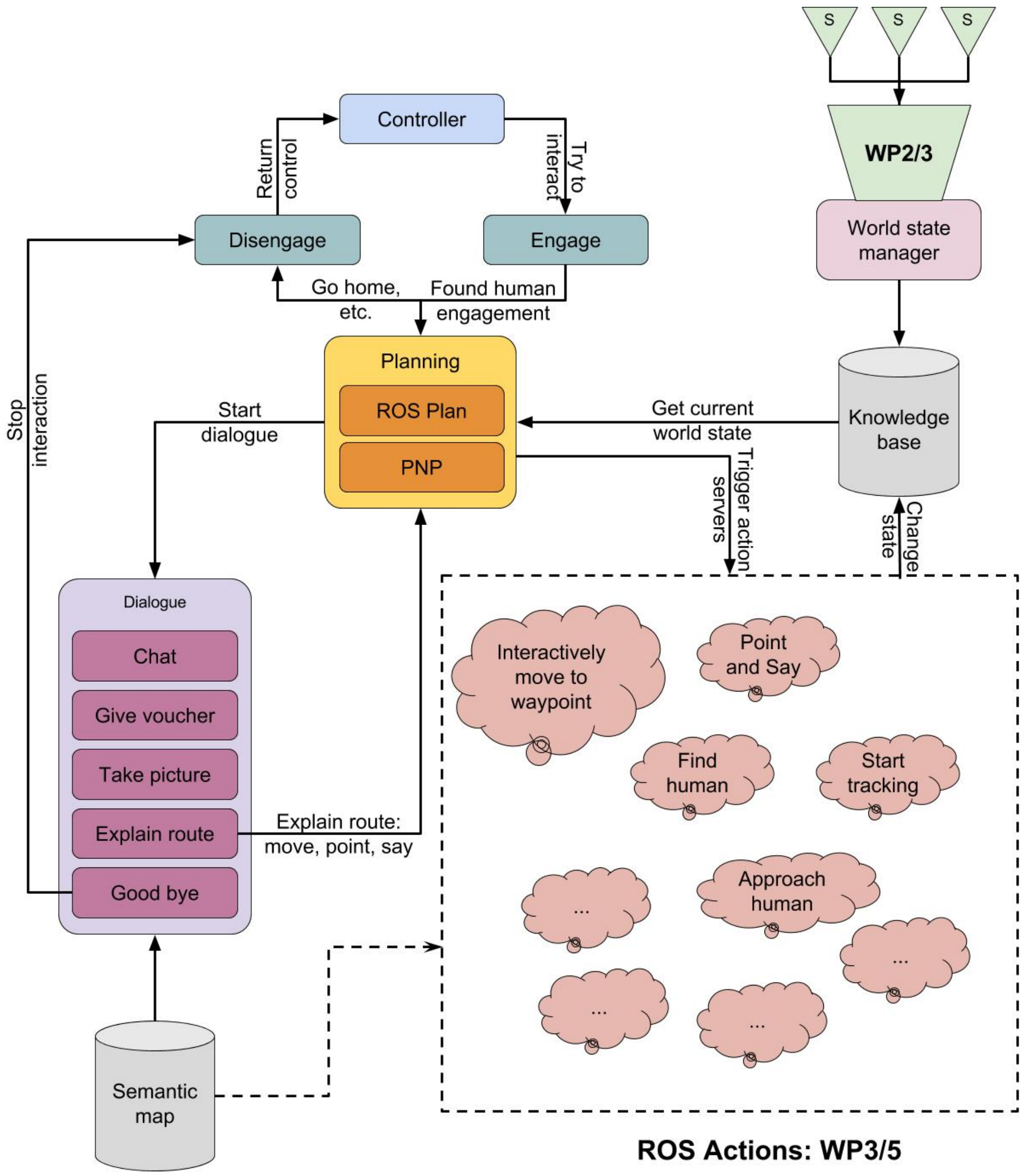


Figure 1: The overall system architecture

2 MuMMER Prototype Components (HWU)

In the system developed at HWU we follow a two-pronged approach to action selection: i) a global planner that is responsible for sequencing actions to achieve a certain high-level goal (see Sec. 2.1), ii) an MDP used during dialogue which either decides to chat to the person or to select a task (see Sec. 2.2). Wherever the task selected by the dialogue is not an atomic action the global planner is invoked to find and execute the right sequence of actions to achieve it.

The source code of the components presented in this section can be accessed through the MuMMER website at `http://www.mummer-project.eu/outputs/software/`.

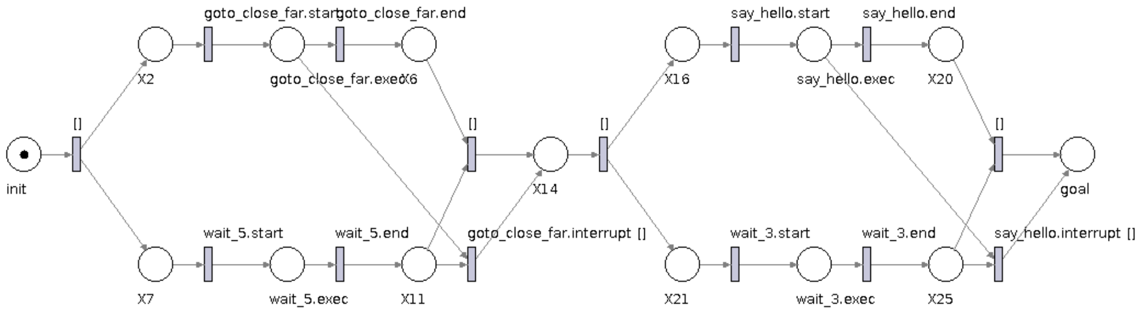


Figure 2: Simplified example PNP for a 'greet' action where the robot approaches the human and then says hello. Both actions have a specified duration which is why a concurrent `wait` task has been added.

2.1 Global Planner

To achieve robust global planning and fast execution of the generated plan, we combined two existing approaches, i.e. ROSPlan [1] and Petri-Net Plans (PNP) [4]. ROSPlan uses a Partial Order Planning Forward-chaining (POPF) approach to action selection which relies on a hard-coded domain representation using PDDL. While the framework offers its own execution of ROS action servers, we opted for replacing this with PNPs to guarantee fast and reliable execution. To this end both components have been altered to allow them to work together where major modifications to the PNP framework have been made to be able to create Petri-Nets online from a generic plan and automatically insert recovery behaviours based on the domain definition to quickly deal with possible changes of the world state unforeseen by the planner. This helps to avoid costly replanning where possible. Other benefits of using PNPs are their ability to handle concurrency (see Fig 2) and to mathematically prove that there are no deadlocks or unused states in the final PNP. To simplify usage, a generic and modular PNP server has been created that allows the seamless integration of standard ROS action servers for partners to develop action execution components while being unfazed by the implementation details of the planning component itself.

In order to be able to test the planning system, several supporting components have been developed that emulate the future results on the other work packages (see Fig. 1).

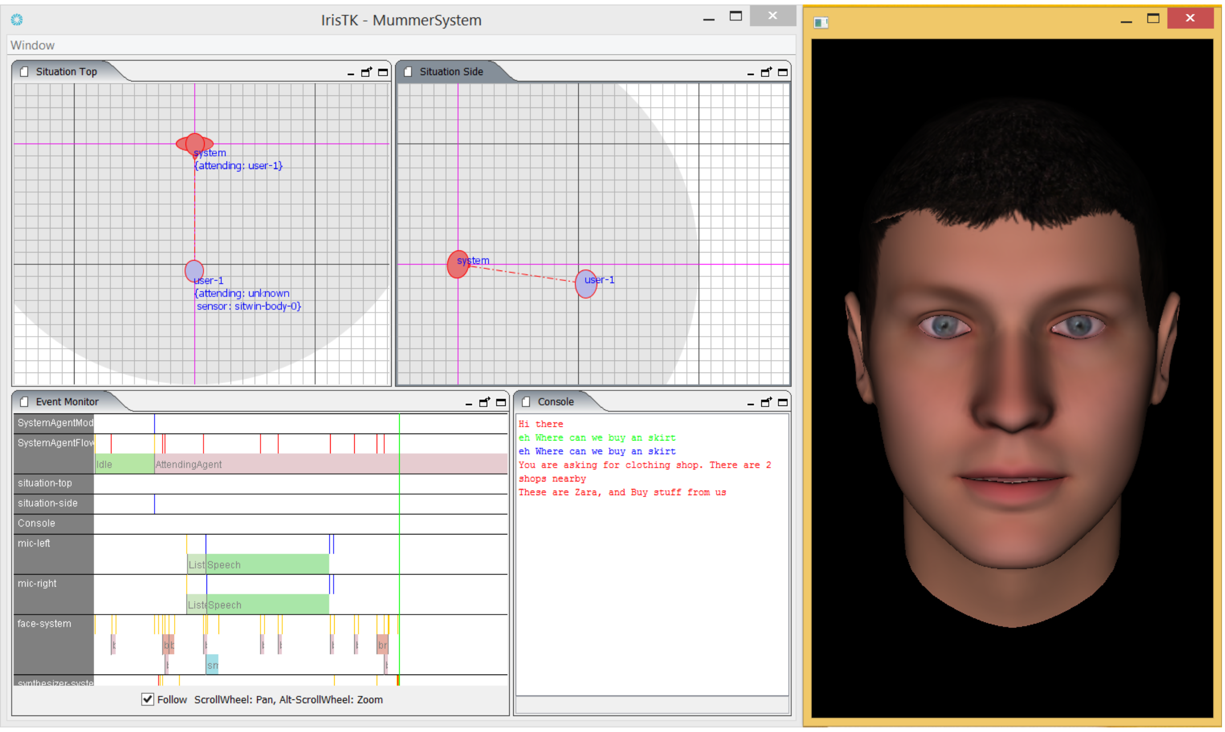
Controller The controller is a simple substitute for a future reward function that decides the behaviour of the robot (see Sec. 4). Currently, there are a couple of high-level goals that the robot could choose, i.e. engage with human in dialogue or give vouchers. These are currently chosen based on an ad-hoc defined probability.

World State Manager In order to allow for any kind of planning, a representation of the current state of the world has to be implemented. For the presented system, logic predicates of the form `(robot_dist id_15 close)` are generated and inserted into the knowledge-base which can be queried during the planning process. This component is able to parse any kind of ROStopic to create such predicates. Moreover, it allows for the automatic generation of qualitative representations like the one mentioned above which describes that the robot is `close` to the person with `id_15`.

Knowledge-Base The HWU knowledge-base (KB) is a simple data base (mongodb[1]) that can easily be accessed via its python or C++ API, or a specialised ROS component to store messages (mongodb_store[2]). Using this form of KB we make sure that the system is robust to errors by storing the current state of the world in a

[1] http://www.mongodb.com/

[2] http://wiki.ros.org/mongodb_store



```
Hi there
eh Where can we buy an skirt
eh Where can we buy an skirt
You are asking for clothing shop. There are 2
shops nearby
These are Zara, and Buy stuff from us
```

Figure 3: Visualisation of the dialogue system using IrisTK. This has since been ported to ROS.

persistent manner. Moreover, the state can be queried at every point in time without having to wait for another component to publish an update.

Dummy Actions To emulate the results of future work of the other work packages, a list of action servers used to, e.g. move, point, speak, track, etc. has been created which predominantly rely on NAOqi functions. As mentioned above, these action servers can easily be replaced by the actual components developed in the other work packages, thanks to the modular design of the planning system.

2.2 Dialogue System

The dialogue component (see Fig. 3) will be presented at the HRI Pioneers workshop at HRI 2017 (see Sec. 6) [2]. To achieve the objective of creating a helpful and entertaining robotic system, we combine chat-style and task-based dialogue which results in a system that can carry out a mixture of task-based and chat-style dialogues, hopefully resulting in a more useful system which is more fun and natural for its human users. However, a problem in combining the two approaches is in creating a Dialogue Manager (DM) which can sensibly switch between task-based and chat-style dialogue in a 'natural' way which users find acceptable. To achieve this, a policy for an MDP has been trained using simulated data from a shopping mall domain.

Speech Input Processing The Automatic Speech Recognition (ASR) module used is Nuance Cloud, running within the NAOqi framework. To determine whether the user has given a task, a semantic grammar is used, which picks various domain related keywords to recognise the task (changing certain task related state variables as mentioned below).

Chatbot For the chatbot we used a modified version of the Rosie[3] bot produced by PandoraBots, using the Program-Y[4] Python implementation for AIML 2.0. The chatbot was adapted to the shopping mall domain, to make it more appropriate for use in this project.

Reinforcement Learning The standard Q-Learning algorithm [3] was used to train the agent, using a simulated user emulating how the users could react to each action the agent takes. For training, the discount factor was set to 0.99, since the agent should care about long-term rewards, while the learning rate was kept fixed at 0.1. In order for the agent to explore as much as possible during the early stages of the training, an ε-greedy policy is followed with an initial ε of 0.9, decaying over time. The system's states, denoting the agent's knowledge about its environment at any given time, is represented with features e.g.: `Mode`, `Distance`, `Timeout`, `LowConf`, `TaskFilled`, `TaskCompleted`, `UserEngaged`, `CtxTask`, `Goodbye`, `UserTerminate`, `TurnTaking`, `PreviousAction`.

The agent is able to select amongst the following 8 actions: `PerformTask`, `Greet`, `Goodbye`, `Chat`, `GiveDirections`, `Wait`, `Confirm`, `RequestTask`. These are converted to text using a mixture of template-based generation and database lookup, and are then synthesised as combinations of speech and robot gestures. The reward function awards each completed task with +10, if the agent greets appropriately +100, and +5 for each consequent turn. It also penalises when the user leaves abruptly with -100.

3 Interaction based Learning Framework for Proactive and Dynamic Dialogue Planning (SBRE)

Work has also been done at SBRE on a complementary system focussing on updating its knowledge-base by learning from humans.

The SBRE system constitutes 4 main components:

Semantic Analyser The SBRE analyser parses a sentence into meaningful entities, like action verb, object, subject, etc.

Context Extractor Tags a sentence in terms of contextual information, which helps the robot to understand context and the situation of the conversation, e.g. if it is a discussion about the weather, or if the person is talking about going out.

Dialogue Module for Knowledge Building Help the robot to ask questions based on the above two analyses, by instantiating proper parameters in the appropriate dialogue branches.

Knowledge-base Updater Module This enables the robot to learn from the above dialogue and enrich the existing SBRE knowledge-base with various information.

The current implementation of the system works in a rule based manner, in the sense that it learns some association of symbols and matches them as described in Fig. 4. The resulting dialogue is then synthesized based on the instantiation of variables of the pre-programmed dialogue patterns.

3.1 Instantiation for Proactive Behaviour of Robot

Context and Situation based Proactive behaviour (interaction and motion) synthesis is a key for Socially Intelligent Human-Robot Interaction. The example illustrated here is to achieve a kind of behaviour where the robot learned during interaction with one human a problem and a potential solution, and used this learned concept to proactively suggest a solution the another human in a similar situation.

[3] https://github.com/pandorabots/rosie

[4] https://keiffster.github.io/program-y/

Example Interaction 1: Acquiring knowledge through interaction.

Max It is so Annoying.

Pepper Is something bothering you?

Max Yes.

Pepper So what is the problem?

Max It is freezing.

Pepper I have no solution for this. I am sorry. I would like to understand, why freezing is a problem.

Max I will get cold.

Pepper Is it possible to solve this?

Max Yes.

Pepper How?

Max By wearing a coat.

Pepper OK, good to know.

Example Interaction 2 Using the acquired knowledge to proactively suggest a solution.

Tina I am freezing.

Pepper In order to avoid getting cold, you should wear a coat.

Tina Thank you.

The application of such a learning based proactive system in a mall like scenario can be useful for many reasons. For example the robot can learn that it was raining, that the discussion corresponds to the weather, that one can get wet in the rain, and to get wet is a problem. The solution to getting wet in the rain is to take an umbrella. The next day the robot can proactively suggest to other visitor (or the same visitor) to take an umbrella if the robot infers that the visitor is going out and that it is raining. This kind of learning behaviour and adaptive dialogue will help to better connect the robot with the visitor by developing trust in the system. One of the desirable side effects and applications for the stakeholders could be the ability to promote of sales of a particular item and brand, e.g. umbrellas and rain coats in the example given above above.

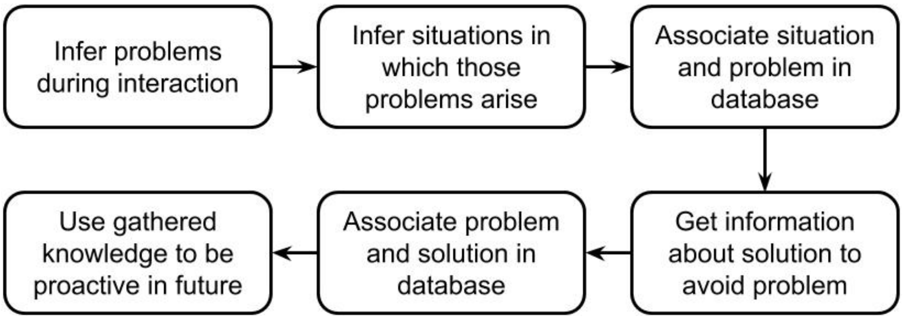

Infer problems during interaction
Infer situations in which those problems arise
Associate situation and problem in database
Get information about solution to avoid problem
Associate problem and solution in database
Use gathered knowledge to be proactive in future

Figure 4: The overview of the interaction based learning framework.

3.2 Future Work

The self learning dialogue will be extended to include action selection in addition to dialogue planning. This action selection and dialogue planning will be integrated with the other planning approaches currently being developed across the work packages in cooperation with the other project partners.

4 Outputs and Future Directions

The HWU system is able to perform all of the tasks that we as a project set out to achieve in the first year scenario. Moreover, the modular design allows to easily replace action execution components with more sophisticated and advanced motion and speech generation and to include more detailed and new sensor data into the KB. Therefore, we have delivered a system on which the future research undertaken in WP4 can be built. We created the foundations of the planning system in a way that allows us to replace, e.g. the currently hard-coded planning by a learned policy for subsequent deliverables without affecting how components are called and, therefore, will not impede work undertaken in other work packages.

As future work, we intend to replace the dialogue MDP with a POMDP-style DM trained from real user data which the prototype system now allows us to collect. Moreover, the chatbot could be replaced with a domain-specific trained chatbot based on actual user data recorded during interactions in the shopping mall or lab-based experiments at partner institutes. Regarding the planning framework, the currently hard-coded domain file and ROSplan which is not able to handle concurrent tasks, will be replaced with a POMDP-style planner trained on real user data. In addition to the input of WP2 and WP3 currently used, this will also include results of Task 5.2 using geometric reasoning.

5 Deviations

There have been no deviations from the work plan.

References

[1] Michael Cashmore, Maria Fox, Derek Long, Daniele Magazzeni, Bram Ridder, Arnau Carrera, Narcís Palomeras, Natàlia Hurtós, and Marc Carreras. Rosplan: Planning in the robot operating system. In *ICAPS*, pages 333–341, 2015.

[2] Ioannis Papaioannou and Oliver Lemon. Combining chat and task-based multimodal dialogue for more engaging HRI: a scalable method using Reinforcement Learning. In *Proc. HRI Pioneers*, 2017.

[3] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[4] Vittorio A Ziparo, Luca Iocchi, Pedro U Lima, Daniele Nardi, and Pier Francesco Palamara. Petri net plans. *Autonomous Agents and Multi-Agent Systems*, 23(3):344–383, 2011.

6 Publications

Combining Chat and Task-Based Multimodal Dialogue for More Engaging HRI: A Scalable Method Using Reinforcement Learning[*]

Ioannis Papaioannou
Interaction Lab, Heriot-Watt University
Edinburgh
i.papaioannou@hw.ac.uk

Oliver Lemon
Interaction Lab, Heriot-Watt University
Edinburgh
o.lemon@hw.ac.uk

ABSTRACT

We develop the first system to combine task-based and chatbot-style dialogue in a multimodal system for Human-Robot Interaction. We show that Reinforcement Learning is beneficial for training dialogue management (DM) in such systems – providing a scalable method for training from data and/or simulated users. We first train in simulation, and evaluate the benefits of a combined chat/task policy over systems which can only perform chat or task-based conversation. In a real user evaluation, we then show that a trained combined chat/task multimodal dialogue policy results in longer dialogue interactions than a rule-based approach, suggesting that the learned policy provides a more engaging mixture of chat and task interaction than a rule-based DM method.

1. INTRODUCTION

Spoken dialogue systems (e.g. [6, 7, 11, 3]) are generally task-based and fail to be engaging for users, concentrating instead on discovering user goals through multiple dialogue turns (such as booking a flight or finding a suitable restaurant). On the other hand, chatbots (such as [1]) are focused on entertainment, and usually do not support execution of user tasks, due to limited memory, and do not perform true language understanding to determine the user's goals. Systems such as *Siri* and the *Amazon Echo* do combine some aspects of chat and task-based interaction, but generally only react to single user turns/commands, and do not support extended multi-turn dialogue to discover user goals.

Combining the two technologies should therefore result in systems which can carry out a mixture of task-based and chat-style dialogues, hopefully resulting in more useful systems which are more fun and natural for their human users. However, a problem in combining the two approaches is in creating a Dialogue Manager which can sensibly switch between task-based and chat-style dialogue in a 'natural' way which users find acceptable. Here, we compare two approaches to this problem – a rule-based method and a policy that is derived via Reinforcement Learning [7, 11].

1.1 Related work

There has been some limited work on combining chat- and task-based dialogue, though none of it has been in the context of HRI. In [10], a text-based hybrid system was implemented, combining a Dialogue Manager with a Chatbot. The system showed promising results, being capable of holding long conversations, but only in *issue-based* dialogues (meaning that *"it views dialogue as the raising and resolving of questions"* [10]).

A hybrid system was also proposed in [2] merging a chatbot with a dialogue manager in a rule-based manner. Their proposed system would have access to both a local and external knowledge base, that along with the user dialogue input would be able to extract the user's goal and his interactions with the environment.

Both of these systems, although somewhat successful in providing semantic representations combined with chatbot conversation, are lacking in terms of extensibility and maintainability. In contrast, systems using Reinforcement Learning (RL) can be trained on data, or via interaction with users, and can learn optimal dialogue policies [7, 11, 4]. Therefore it is of interest to determine whether combined chat/task multimodal dialogue systems for HRI can also be trained using RL methods.

Note that these prior systems do not use multimodal information to enrich the dialogue, as is required for HRI. Our system uses distance information (provided by a Kinect sensor), but RL can also optimise action selection dependent on a wider variety of sensory inputs.

2. SYSTEM COMPONENTS

Two versions of the hybrid (chat+task) system have been developed and evaluated, in the context of a robot which can help and entertain visitors to a shopping mall. The first acts as a baseline approach, where the actions are decided using handcrafted rules, and the other uses actions learned through RL. The tools used can be broken down to 3 individual but intertwined software modules:

- *Program AB* [1], which runs the chatbot, see section 2.2;
- *BURLAP* [5], a RL framework to train the MDP policy, section 2.3;
- *IrisTK* [8], integrates the subsystems, as well as handling *speech recognition* and *speech synthesis*.

[*] This research is supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 688147 (MuMMER project http://mummer-project.eu/)

HRI '17 Companion March 06-09, 2017, Vienna, Austria
© 2017 ACM. ISBN 978-1-4503-4885-0/17/03…$15.00
DOI: http://dx.doi.org/10.1145/3029798.3034820

2.1 Speech Input processing

The Automatic Speech Recognition (ASR) module used is Nuance Cloud, running within the IrisTK platform. To determine whether the user has given a task, a semantic grammar is used, which picks various domain related keywords to recognise the task (changing certain task related state variables as shown in section 2.3).

2.2 Chatbot

The chatbot was implemented using *Program AB* [1], which is a Java implementation for the AIML 2.0 specification. A predefined sample bot called *S.U.P.E.R.* was altered in order to be appropriate for a shopping mall domain, for example informing the user of mall opening times.

2.3 Reinforcement Learning

The standard *Q-Learning* algorithm [9] was used to train the agent, using a simulated user emulating how the users could react to each action the agent takes. For training, the discount factor was set to 0.99, since the agent should care about long-term rewards, while the learning rate was kept fixed at 0.1. In order for the agent to explore as much as possible during the early stages of the training, an ϵ-*greedy* policy is followed with an *initial* ϵ of 0.9, decaying over time.

The system's *states*, denoting the agent's knowledge about its environment at any given time, is represented with features e.g.: *Mode, Distance, Timeout, LowConf, TaskFilled, TaskCompleted, UserEngaged, CtxTask, Goodbye, UserTerminate, TurnTaking, PreviousAction.*

The agent is able to select amongst the following 8 actions: *PerformTask, Greet, Goodbye, Chat, GiveDirections, Wait, Confirm, RequestTask.* These are converted to text using a mixture of template-based generation and database lookup, and are then synthesised as combinations of speech and robot gestures. The *reward function* awards each completed task with +10, if the agent greets appropriately +100 and +5 for each consequent turn. It also penalises when the user leaves abruptly with -100.

3. EVALUATION

To evaluate the effectiveness of the hybrid system, we compared it against versions of the system with only chat-based features or task-based features enabled. The results in fig. 1 show how the hybrid system managed to get a higher average reward compared to the chat/task-only versions.

Moreover, we evaluated the system with real users, with 10 subjects that were introduced to the RL trained version, as well as a version using only hand-crafted rules for action selection. The users of the RL system had longer dialogues, on average (136.8 versus 112.8 seconds), indicating that this system is more engaging.

4. CONCLUSION

A hybrid Conversational Agent for HRI was implemented, for the first time combining a task-based Dialogue Manager with a Chatbot. The system also uses multimodal sensors (distance, using a camera system), provides multimodal output with an animated face, and uses an action-selection policy trained using Reinforcement Learning (RL). The system acts based on a trained MDP policy, and was evaluated against a version of the same system using a handcrafted policy. We also evaluated the benefits of a hybrid system, over those which can only chat or perform tasks (section 3).

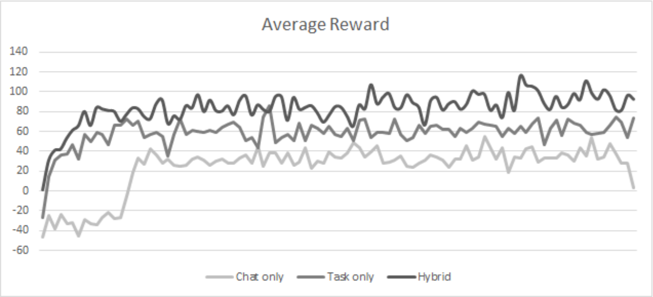


Figure 1: Average total discounted reward per dialogue with simulated users

4.1 Future Work

More sensory input could be provided, such as basic emotion classification,and head pose estimation, while simultaneously substituting the MDP model with an *POMDP*, to accommodate uncertainty in sensing [11]. The increasing complexity of such a system would make also hand-crafting of the interaction rules infeasible, so that an RL method as presented here would be an increasingly attractive solution.

We are currently applying this research to the MuMMER Project (http://mummer-project.eu/) on the Pepper robot where it will be evaluated with real users.

5. REFERENCES

[1] ALICE AI Foundation. Program AB. https://code.google.com/p/program-ab/, 2013.

[2] A. Dingli and D. Scerri. Building a hybrid: Chatterbot - dialog system. *Text, Speech, and Dialogue*, pages 145–152, 2013.

[3] O. Lemon, A. Bracy, A. Gruenstein, and S. Peters. A multi-modal dialogue system for human-robot conversation. In *Proc. NAACL*. 2001.

[4] O. Lemon and O. Pietquin, editors. *Data-driven Methods for Adaptive Spoken Dialogue Systems.* Springer, 2012.

[5] J. MacGlashan. Burlap. http://burlap.cs.brown.edu/, 2016.

[6] M. F. McTear. Spoken dialogue technology: enabling the conversational user interface. *CSUR*, 34(1):90–169, 2002.

[7] V. Rieser and O. Lemon. *Reinforcement learning for adaptive dialogue systems.* Springer, 2011.

[8] G. Skantze and S. Al Moubayed. Iristk: a statechart-based toolkit for multi-party face-to-face interaction. *Proc. ICMI*, 2012.

[9] R. Sutton and A. Barto. *Reinforcement learning.* MIT Press, 1998.

[10] A. van Woudenberg. *A Chatbot Dialogue Manager.* PhD thesis, Open University of the Netherlands, 2014.

[11] S. Young, M. Gasic, B. Thomson, and J. D. Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.