Proactive Communication *in* Human-Agent Teaming

Exploring the Development of Agents that Learn to Proactively Communicate their Observations and Plans using Context Factors in Human-Agent Teaming

MSc Thesis - Artificial Intelligence

Emma M. van Zoelen
6027970

In collaboration with TNO

Supervised by dr. F. P. M. (Frank) Dignum, dr. J. (Jurriaan) van Diggelen,
dr. A. H. M. (Anita) Cremers and dr. M. M. (Marieke) Peeters

Emma Maxime van Zoelen
Proactive Communication in Human-Agent Teaming
Exploring the Development of Agents that Learn to Proactively Communicate their Observations and Plans using Context Factors in Human-Agent Teaming
December 2018

Reviewers: dr. F. P. M. (Frank) Dignum and prof. dr. M.M. (Mehdi) Dastani
Supervisors: dr. J. (Jurriaan) van Diggelen, dr. A. H. M. (Anita) Cremers and dr. M. M. (Marieke) Peeters

Utrecht University
Intelligent Systems Group
Department of Information and Computing Sciences
Princetonplein 5
3584 CC Utrecht

TNO, Nederlandse Organisatie voor Toegepast Natuurwetenschappelijk Onderzoek
Department of Perceptual and Cognitive Systems
Kampweg 55
3769 DE Soesterberg

Abstract

Everywhere around us machines are handed more responsibilities, because with their unique abilities, they are able to outperform humans on many tasks. However, since humans have their own unique abilities in which they still outperform machines, a logical step would be for humans and machines to collaborate in human-agent teams. For such collaboration, it is essential that they communicate smoothly. Communicating as a team member is a difficult challenge, as it requires both humans and agents to be context-sensitive and proactive.

In this thesis, an attempt was made at developing agents that learn how to communicate proactively. A combination of both data-driven learning methods as well as rule-based agent technology was used, while carefully reflecting on the influences of both methods on the learning process and the resulting behavior. Different kinds of learning agents were evaluated both in simulation as well as in an experiment where they worked together with humans. Agents learned to communicate proactively reasonably well after training in simulation, being able to use a minimal amount of communication to improve their performance as much as possible. In transferring to a context in which they played with humans, they were able to use the behaviors learned during simulation, while also learning some additional behaviors. The human team members generally trusted their agent companions, while differences in the extent to which people felt they collaborated with the agent as a team and usability were found for different kinds of communication.

This work is an attempt at bridging the different kinds of research that exist into team communication, by looking into computational and technical methods for developing proactive agent communication while constantly keeping human team members in mind.

Acknowledgements

Working on this project was inspiring, interesting, exciting and fun, but a real challenge at times. As much as I enjoyed working on the topic of making Humans and Agents collaborate, it was not always easy to see the next step. Several people helped me to find the best thing to do next while keeping my head up and enabled me to finish this thesis with pride.

I would like to first and foremost thank my supervisors at TNO, *Jurriaan van Diggelen*, *Marieke Peeters* and *Anita Cremers*. Thank you for the inspiring discussions in the beginning, the critical and extensive feedback towards the end and all meetings in between. Your sometimes contradicting points of view helped me to stay critical and make the most out of the project.

Frank Dignum, I would like to thank you as well for wanting to supervise me even though you did not have much time. Although we did not have much contact during the second half of the project, your feedback in the beginning of the process always helped me forward and your work inspired me to look at social context and communication differently.

All people at TNO that I collaborated with during the MMT-sprints deserve some thanks as well. You helped me to see where my work fit in with the rest of the work done at TNO and helped me understand the bigger picture.

Of course great thanks goes out to *Vincent Koeman*, who helped me with getting Blocks World 4 Teams to work and all small and big problems that I had with it. Every time I emailed you I wondered what you would think, another question, another problem, but you helped me tirelessly every time. Without your help, I sometimes think I would still be lost in the code now!

Special thanks goes out to my fellow interns at TNO, both the group before the summer and the group after. You made my time at TNO more fun, with the many random conversations, games of table tennis and Jong TNO activities that we went to. I wonder where all of us will end up some day!

To conclude, I would like to thank my friends and family for always helping me relax outside of work. And of course *Joris*, thank you for being there for me, for cooking for me when I was busy and came home late, for pushing me to take enough breaks towards the end and for making me laugh. Without those laughs it would have been much more difficult to stay positive and keep up.

Contents

1. Introduction

Every day, machines are becoming capable of handling tasks of growing complexity. As a result, it is inevitable that responsibility for the execution of such tasks is gradually shifting from humans to machines. However, humans have their own unique abilities, such as dealing with ethical issues, emotions and uncertainty, and they are generally unwilling to let go of their responsibility. Therefore, it is key that these machines are capable of working together with humans fluently, as this would enable them to make optimal use of the capabilities of both parties. The ultimate goal would be to have teams of humans and machines collaborating in a similar way to humans collaborating in teams. Such teams would then be called human-agent teams.

Research on human-agent teaming has existed for a while, especially in safety-critical contexts such as space applications (Bradshaw et al., 2003; Diggelen, Bradshaw, Grant, Johnson, & Neerincx, 2009), the military (Parasuraman, Barnes, Cosenzo, & Mulgund, 2007) and search and rescue problems (De Cubber et al., 2012; Kruijff et al., 2014; Mioch, Peeters, & Neerincx, 2018; Nourbakhsh et al., 2005). It is a complex domain with many possible research angles, which is why several people have defined necessary factors and aspects for successful human-agent teaming. A recurring theme in the literature about such factors is communication. Communication is a necessary skill for enabling collaboration and related aspects such as coordination of tasks and maintaining situational awareness (Klein, Woods, Bradshaw, Hoffman, & Feltovich, 2004; Lohani et al., 2017). As humans we intuitively know the importance of communication in teamwork, but we just as well know how hard it can be to communicate effectively and efficiently. A proactive, context sensitive and dynamic type of communication is necessary, that takes into account the needs of team members as well as the needs of the task (Klein et al., 2004). If this is done well by agent team members, apart from improving the execution of the team task, they are considered more reliable and they are accepted more easily as communication partners in general (Lohani et al., 2017).

Existing work on communication systems for human-agent teaming generally does not deal with such proactivity and anticipation and leaves most of the initiative up to the human team member. Work has been done on context-sensitive communication systems (Sordoni et al., 2015), but this serves mostly to personalize messages and incorporate context in replies to human conversation partners. Studies that focus on proactivity either only look at conversational information to determine whether initiative should be taken (Butchibabu, Sparano-Huiban, Sonenberg, & Shah, 2016) or focus on initiative in a conversation that is ongoing, where both partners are fully engaged in the conversation for the total duration of the task execution (Chu-Carroll, 2000). In reality, initiating conversation is especially important in tasks for which conversation is helpful, but not the main activity. Being proactive and initiating communication while taking into account context factors outside of the conversation has not been researched before in such tasks.

It is clear that proactive and context-sensitive communication is vital in teamwork, while it has not been studied much. In this thesis, proactive communication for human-agent teaming is explored, in order to create communication that enables the collaboration process. Proactive in this context means that the agent is able to initiate communication on its own without being prompted by an agreement or utterance made by a human component. Related to this, the agent should be sensitive as to when communication about a certain piece of information is appropriate, given the context.

1.1 Use Case and Context

In order to research proactive communication, it is essential to find an environment as well as a task that can be used for testing and exploration. The task environment in which the current study is conducted is inspired by military use cases. This is a context that is both safety-critical and time-pressured, meaning that team members have to communicate with each other at appropriate times, while there might not always be enough time to communicate. In the future, teams working on such tasks will consist of intelligent machines and humans, in which the machine team members must learn when and what to communicate to their human team members. This boils down to knowing when to take the human team member in the loop, and when to leave them out of the loop to take advantage of the autonomous capabilities of the machine.

The following is an example of a possible conversation within such a context. It is part of a scenario where an unknown car that might be a threat is approaching a protected compound. In this case, the communication is initiated by the machine team member (M), but the human (H) takes initiative as well by asking questions.

M: Human, I see a car that might be a threat.
H: Why do you think it is a threat?
M: It has no license plate.
H: Can you show me the car?
*M: *shows surveillance video of car**
H: How far away is it from our compound?
M: About 5 kilometers.
H: That is quite far away, don't worry about it for now.
…
M: The car is now 3 kilometers away and approaching us with a speed of 80 km/h.
H: That changes the situation. Can you give me a closer look of the car?
*M: *goes a little closer and zooms in at the car* Going any closer than this would be unsafe for me.*
H: That is okay. Send some guarding cars to the fence, tell them to watch out for this vehicle.

As can be seen from this example, within such safety-critical and time-pressured contexts there are many possible situations in which the system is supposed to initiate a conversation, while being sensitive to the desired level of communication that the human team member has. The fact that the human team member asked questions might be an indication that the machine could have given more information at first. However, the fact that no action was taken in the first part of the conversation might be an indication that it would have been better for the machine to wait and only initiate communication when the car would be closer to the compound. On the other hand, the sole purpose of the communication might have been to let the human know that the machine was doing its task. All these subtle factors play a role in when communication should be initiated.

Of course this is merely an example to give an idea of the complexities and subtleties of the problem of proactive communication in general. Communication in teamwork can have many different aims, where this thesis will focus mostly on the aims of informing other team members of observations and plans, while trying to perform as best as possible on the time-pressured task at hand. For the environment in which such a task can be implemented the choice was made to use the Blocks World 4 Teams environment, which has specifically been built for research into collaboration in human-agent teaming (M. Johnson, Jonker, Van Riemsdijk, Feltovich, & Bradshaw, 2009). BW4T is a simple blocks world environment in which agents and humans can collaboratively solve a task. In BW4T, the team members need to search and deliver colored blocks, while performance is measured by the time it takes to deliver a certain sequence of blocks.

1.2 Research Aim and Questions

As stated above, a lot of research on communication systems in general exists and while some work has been done on proactive communication within the domain of human-agent teaming, there are still many opportunities for research. The focus in existing literature is mostly on creating one optimal strategy for communicating within the task at hand, where the goal is to eventually optimize task performance. Related to the context at hand we might say that optimizing for one task is impossible, as the environment is ever changing and not very predictable.

Also, in the process of teaming, task performance is not the only important measure. More subtle and qualitative factors like trust are especially relevant to make the collaboration a better experience for the human part of the team. Preferred levels of autonomy and proactivity for the system might change over time depending on for example the level of trust. The system will have to adapt to its user, while the user adapts to the system.

Following from this, the research question for the current study will be the following:

RQ *Can agents learn to communicate information about observations and plans proactively in order to improve human-agent team performance in the Blocks World 4 Teams environment, with the use of context information?*

As derived from existing evaluation metrics and methods, team performance in this context is first and foremost performance on the task at hand. However, in order to obtain more detailed insights, dialogue efficiency, trust and usability will be considered part of team performance as well.

Apart from the main question, there are several relevant sub questions. Since the context asks for an adaptive agent, but the safety-critical environment demands a system that makes little mistakes, a combination of data-driven learning methods and rule-based methods was implemented that attempted to balance the two to make use of their strengths. The aim was to enable learning within a relatively low number of attempts. This leads to the first sub question:

SQ1 *How can data-driven learning methods be balanced with rule-based methods to achieve fast learning of useful proactive communication behavior?*

Since there are many ways and configurations in which it is possible to implement the abovementioned methods, it is relevant to look at how those different configurations influence the learning process and outcomes. Especially since one of the main challenges in communication systems in general is that it is very hard to extract feedback from communicative acts, it is relevant to see how different rewards and state representations can support the learning process. The second sub question therefore is the following:

SQ2 *What is the influence of state representation, exploration and reward on the learning process and learning outcomes of proactive communication strategies?*

To evaluate whether the learning agents actually achieve a good performance, it is necessary to evaluate them thoroughly, in simulation as well as while playing with humans. There is a large difference in the way communication systems are evaluated in technical literature, as opposed to literature from the fields of psychology or human-computer interaction. To be able to evaluate the agents thoroughly, the third sub questions is as follows:

SQ3 *How do we measure team performance in the context of human-agent teaming in BW4T?*

Several existing methods and metrics for measuring team performance have been reviewed. Combining and integrating the most promising ones will allow for the measuring of team performance as a whole.

Apart from creating overall improvement of team performance, it is relevant to be able to identify which aspects of the communication influence which aspects of team performance, such that this can be taken into account in the design of future communication systems. Also, since agents were designed in simulation, but had to perform when playing with humans, it had to be evaluated how learned behavior transfers from simulation to a human-agent context. This leads to the following two sub questions:

SQ4 *How does proactive sharing of information about observations and plans influence team performance in a human-agent team?*

SQ5 *To what extent is an agent able to transfer learned proactive communication behavior from a simulation environment to an environment where it plays with humans?*

Using the BW4T environment as a basic task environment, agents were developed that aim to learn how to communicate proactively. These agents were evaluated in an experiment where they played with human team members. Chapter 2 gives an overview of the literature related to the challenge of proactive communication, while Chapter 3 gives a detailed description of the approach taken. In Chapters 4 and 5, the development of agents that learn how to communicate proactively is discussed. The experiment in which these agents were evaluated and its results are discussed in Chapter 6. An extensive discussion of both implications of the results as well as limitations of the methods chosen is given in Chapter 7, while the conclusions drawn from these results are summarized in Chapter 8 and opportunities for future work are presented in Chapter 9.

2. Literature Review & Related Work

This chapter provides an overview of literature that relates to the topic of learning to communicate proactively in a teaming context. It crosses different disciplines, such as organizational psychology, human-computer interaction, computer science and artificial intelligence.

2.1 Communication in Human Teams

When trying to create good and efficient communication between humans and machines in teams, human-only teams can be used as a starting point. More specifically, there are many studies on the dynamics of human-only virtual teams, in which human team members do not know each other before the team process and will not meet physically (He, Butler, & King, 2007; Henttonen & Blomqvist, 2005; Jarvenpaa & Leidner, 1999; S. D. Johnson, Suriya, Yoon, Berrett, & La Fleur, 2002). In these studies usually all communication is done via a web interface, allowing for a controlled environment that maps quite well to the human-agent team scenario. The communication is often restricted to text messages in the existing literature, meaning that only the task performance and these text messages can be used to build a functioning team. This is basically the same in human-agent contexts; humans and agents can only communicate via a digital interface of some sort, within which the actual interaction can be multi-modal.

2.1.1 The Teaming Process

Since teams are not formed as well-functioning and synergetic teams immediately, in human-only teams a period of time in which the team develops itself is inevitable (Jarvenpaa & Leidner, 1999; S. D. Johnson et al., 2002; Warkentin & Beranek, 1999). Team members have to get used to each other's way of working as well as their communication patterns in several phases. They will have to get to know each other's personality and abilities in the first phase. During the second phase, conflicts may arise and team members need to deal with the conflict by developing task and social structures. These might include consensus about communication patterns. In the final phase the task can actually be performed efficiently, as the team then knows how to work well efficiently. While many people first wrote theories about this process, it was also empirically validated (S. D. Johnson et al., 2002). Adapting to the personalities and capabilities of team members over time can be seen as the starting point for creating a well-functioning team and will be regarded as an important aspect throughout the rest of this thesis.

During this process, a challenge that exists specifically in the context of the virtual team is the development of social relationships. This is important in teams and increases team performance, but it is harder to develop such social factors without face-to-face meetings (Jarvenpaa & Leidner, 1999; Warkentin & Beranek, 1999). To be more precise, while it is possible to grow strong relational links in virtual teams, the process is much slower. Still, using social communication to ensure social bonds in virtual teams might be even more important than in regular teams as it greatly increases trust among team members (Jarvenpaa & Leidner, 1999). While this will at some point become relevant in human-agent teaming as well, it is not actively integrated in the current work, since creating a social bond with an agent can be seen as a separate research challenge.

2.1.2 Communication Strategies

Next to communication that supports social relationships within the team, one might wonder specifically what kind of communication patterns enable a smooth teaming process and a good task performance. Frequency of communication has been used as an indicator of team performance, where a higher frequency of communication indicates that the team does better at cooperating and performing the task (He et al., 2007). It is however also mentioned that too much communication can lead to attentional overload, leading to distraction and lower cognitive focus, which might actually decrease task performance (Leenders, Van Engelen, & Kratzer, 2003). Thus a balance must be found in the amount of communication used within the team. Related to that, the best performing teams use mostly implicit communication and much less explicit communication (Butchibabu, 2016). The term implicit communication is defined as (verbal) communication done after team mates anticipated the communication need of their team members. In contrast, explicit communication consists of (answers to) requests for information. Using implicit communication can reduce the total amount of communications necessary, making it less likely that attentional overload will occur.

The concept of implicit communication can be divided in two categories, to define even more specifically what type of communications are beneficial to a teaming process: deliberative-implicit and reactive implicit communication, where deliberative-implicit communication relates to information about the actions of a team member (e.g. 'I am doing this task'), whereas reactive-implicit communication relates to the state of the world or team member (e.g. 'There is useful information here'). It was found that high performing team members use relatively more deliberative-implicit communication, suggesting that this might enable team members to create better mental models of their partners and decrease the necessary amount of communication even further (Butchibabu et al., 2016).

Next to the effect of specific strategies as mentioned above, some general values for communication in teams have been identified. These include punctuality, active participation and timely responses (S. D. Johnson et al., 2002). Those values together emphasize the importance of time and understanding the right time for taking initiative to communicate, for which it is necessary to understand the current context as well as the information needs of team members.

2.1.3 Human Teams as a Starting Point

Looking at the above insights, it can be concluded that in order to create an effective team, it is important that team members get to know each other and have the ability to adapt to each other's way of working. In terms of communication, the right balance must be found between enough but not too much communication, by anticipating the communication needs of team members and communicating implicitly most of the time. In this, timing and punctuality are values that play an important role. For this thesis, adaptivity and balancing out the right amount of communications at exactly the right time have been taken as the main challenge. In order to make agents team members that are just as good as human team members, it has been attempted to make them learn specifically this. Above, social bonds between team members have been mentioned as an important factor as well, but these will not be considered in this work.

2.2 Algorithms for Human-Agent Communication

Algorithms for communication as well as algorithms for dialogue have a long history of research and development. In order to understand how human-agent communication might be implemented in an adaptive and proactive way, an overview of different existing technologies and algorithms is given. Each have their advantages and disadvantages related to the type of communication that is desired within a given context. The focus is for a large part on dialogue systems, while the work done in this thesis is not aimed at creating a dialogue system. However, looking at human-agent communication systems and algorithms, dialogue systems are a large part of the existing research. Also, proactive communication deals with coming up with a decision about communication based on context, while dialogue systems deal with coming up with a decision about a reply based on the previous message, which can be considered a special kind of context. This means that the basic problem has many similarities.

2.2.1 Dialogue Technology

The research on computational dialogue and dialogue systems is extremely diverse. Many chatbots exist on the internet that can decently hold a conversation, but these chat-agents often make mistakes. The types of mistakes they make, however, are related to the type of chatbot that is being dealt with. They can either be designed for conversational purposes, or to help humans complete a task on a website. In the literature, the first type of dialogue is called chit-chat and the latter is usually called task- or goal-oriented dialogue (Yan, 2017).

Agents designed for chit-chat dialogue are designed to keep a conversation going, to be funny, to ask general questions fitting to a given conversation; in general, to entertain. Agents designed for task-oriented dialogue, however, usually have a very clear domain-specific goal. Think for example of a chatbot that can help you book a table at a restaurant, or one that can help find out how much money you will get from your health insurance for a certain medical treatment. In the current study, we will mostly be dealing with task-oriented communication. In the specified context of human-agent teaming, humans and agents will be collaborating to achieve a common goal. This context can therefore imply two tasks towards which the communication can be oriented, namely the achieving of the goal as well as the task of maintaining a good collaboration.

Slot Filling: finite-state automata

Traditional task-oriented dialogue systems as well as most current task-oriented chatbots make use of a technique called slot filling. An example of such a system has been made by Google ("Dialogflow," 2018), which is one of the most advanced chatbot building tools that is currently on the market. Systems like these are based on finite-state automata, where the dialogue space consists of several dialogue states (Goddeau, Meng, Polifroni, Seneff, & Busayapongchai, 1996). In chatbot building tools, these states are usually called 'intents'. While in dialogue, the agent tries to recognize or classify which intent the user has from the utterance made. Intents are predefined by the programmer, and represent a range of potential utterances from the user corresponding to a particular type of action by the agent (e.g. a direct response, API call, etc.). Intent classification can be done via a rule-based system, or through machine learning.

The system described in Goddeau et al. (Goddeau et al., 1996) makes use of an E-form to complete the task, but this is basically a synonym to the slot filling technique. Their E-form is a virtual form that contains slots for relevant information about the domain or task. In order to complete the task at hand, the slots have to be filled with information that the user provides in dialogue. To fill the slots, the agent will have to extract the information from the user's utterances. The entities containing this information are defined in the intents to make sure the agent will be able to find them.

Intents as well as actions or responses related to the intents have to be hand-coded, which is time consuming and cumbersome. Because of this, the resulting dialogue system can only be used for very narrow domains and is not flexible in general. However, slot filling systems are quite reliable in many narrow-domain applications, which is why they are still widely used, for example for chatbots. The robustness is an advantage that was attempted to be integrated into the agents developed in this thesis.

Probabilistic Dialogue: (PO)MDPs

In order to make traditional dialogue systems more flexible and less time-consuming to implement, research has investigated the use of probabilistic methods in dialogue systems. The key processes that a dialogue manager engages in are still (a) tracking the state of the dialogue and (b) planning an action or policy based on that state. Yet in contrast to the slot-filling version, a probabilistic dialogue manager models the dialogue as a Markov decision process (S. J. Young, 2000) using reinforcement learning, as this enables it to optimize the dialogue policy for a particular task. More specifically, many studies on dialogue systems work with partially observable Markov decision processes (POMDPs) (J. D. Williams & Young, 2007; S. Young, Gašić, Thomson, & Williams, 2013). Such models combine belief state tracking (as the state the user is in is uncertain) and reinforcement learning for optimizing the dialogue policy. According to Young et al. (S. Young et al., 2013), this has several advantages. First of all, uncertainty about the state is explicitly modeled and updated in a Bayesian manner. Due to this, one error has significantly less impact than in a finite state machine; if an utterance is repeated, the system's belief in the content will eventually increase, repairing a possible error. Secondly, since a belief distribution over all possible states is kept, when an error occurs, the system can simply change the probabilities and switch to another state. No explicit error strategies are necessary. Last, the explicit representation of both state and policy-derived action allows for adding rewards to state-action pairs, to be able to incorporate criteria for the dialogue. This makes it much easier to optimize the dialogue, and does not require a lot of manual tuning.

There are however several disadvantages to the use of POMDPs as well, which is probably why the older slot filling mechanisms are still often used. While part of the reason to use POMDPs is the fact that it optimizes dialogue policy without having to hand-code all the rules, the policy optimization for real-world dialogue takes a very long time to complete, as it needs a lot of data to converge. Therefore, training with real users is practically impossible, and systems are usually trained with user simulations. Related to that, the optimization works with a reward system, but it is hard to extract realistic and reliable dialogue-related rewards from a real user. According to Young et al. (S. Young et al., 2013), these problems must be solved before dialogue systems using POMDPs can reliably be implemented. However, the flexibility and adaptivity of probabilistic communication is an aspect that is very valuable for the creation of proactive communication.

Neural Networks (Memory Networks/Neural Turing Machines)

More recently, people have tried to create dialogue using Recurrent Neural Networks (Ritter, Cherry, & Dolan, 2011; Serban, Sordoni, Bengio, Courville, & Pineau, 2016; Shang, Lu, & Li, 2015; Sordoni et al., 2015; Vinyals & Le, 2015). A technique called sequence-to-sequence (seq2seq) or encoder-decoder modeling is used to create such dialogue systems. This means that the model can use a sequence of text as input, while the output will also be a sequence of text; this makes the model end-to-end trainable. It can therefore be trained using just large amount of dialogue data obtained from for example social media, without having to explicitly model or annotate that data.

Most of the beginning work on data driven dialogue systems is, however, on chit-chat. The system aims to 'decode' or 'translate' an utterance into a response that is correct, but not necessarily meaningful. While Sordoni et al. (Sordoni et al., 2015) do incorporate context-sensitivity to improve the dialogue quality, their dialogue is only evaluated on quality of the response itself, and not on the ability of the system to perform conversational tasks.

Several people have therefore started working on so-called 'Memory Networks' (Sukhbaatar, Szlam, Weston, & Fergus, 2015; Weston, Chopra, & Bordes, 2014) or 'Neural Turing Machines' (Graves, Wayne, & Danihelka, 2014). These are Neural Network models extended by a long-term memory that can be used as a knowledge base. The algorithm can learn how to use this knowledge base depending on the task. While originally this was a general algorithm architecture, Weston et al. (Weston et al., 2014) already focused on textual output and evaluated their algorithm on a question answering task. Dodge et al. (Dodge et al., 2015) evaluated several end-to-end trainable models on different dialogue tasks, such as question answering and recommendation dialogue. From this evaluation, it became clear that while all models perform reasonable, Memory Networks gave the most promising result overall.

An attempt to test Memory Networks in a goal-oriented setting has been made by Bordes et al. (Bordes, Boureau, & Weston, 2017). In their work, they compare the algorithm to a hand-crafted slot-filling baseline, to test whether these data-driven and more generalizable models can compete with traditional narrow-domain dialog models. They show that while Memory Networks perform almost as well as rule-based slot-filling algorithms on a per-response measure, the per-dialogue measure for some tasks is still very low. However, these scores all rely on synthetically generated language. In a more realistic task based on human-bot dialogue, the Memory Network outperforms the traditional model. However, some of the problems that exist in POMDPs, such as slow convergence that needs a lot of data and the challenge of extracting the right user feedback still exists when implemented with a reinforcement learning mechanism.

Social Practices

An alternative way to determine the right communicative act in a dialogue has been built around the idea of Social Practices (Augello, Gentile, Weideveld, & Dignum, 2016; F. Dignum & Bex, 2018). The term Social Practice comes from sociology, where it is used to define the context of a situation. In order to use it as a definition for context in agent systems, it has been defined in terms of physical context (resources, places, actors), social context (social interpretations, roles, norms), activities, plan patterns, meaning and competences (V. Dignum & Dignum, 2015). In the creation of dialogue, these context factors limit the amount of communicative actions that an agent can do, enabling agents to quickly make context based decisions if the situation is simple. If the context is more complicated, the social practice creates a subset of possible actions, and another deliberation process can be chosen to make a final choice between the possibilities. In the current study, it is attempted to have agents learn proactive communication with the use of context factors as well. While this context has not specifically been defined as a Social Practice, the concept has been an inspiration for the general approach taken, in which agents attempt to learn to be sensitive to social contexts.

2.2.2 Agents Learning Communication for Collaboration

In the abovementioned dialogue technologies agents are mostly preprogrammed, learned in a passive and supervised way by exposing them to large amounts of dialogue text or learned only in a conversation-related context. Lately several papers have been published that take a different approach. The authors of these papers argue that especially when communication is necessary for collaboration, supervised approaches do not capture the subtle ways in which humans use communication to coordinate in collaborative tasks (Lazaridou, Peysakhovich, & Baroni, 2017). Their methods and ideas build on research in which agents learn to coordinate their actions in collaborative games, based on repeatedly playing a coordination game and receiving some utility for the chosen action (Kapetanakis & Kudenko, 2002).

Coordinating actions and using communication to coordinate actions are however different in the sense that communication adds a layer of complexity on top of merely selecting actions. Basically, communication can be seen as separate actions that must be coordinated as well, which might be costly when resources are limited. Some models based on Markov decision processes have been developed, that tried to enable agents to learn a trade-off between the cost and value of communication actions, to optimize a communication policy in a given game (Goldman & Zilberstein, 2003; Xuan, Lesser, & Zilberstein, 2001). This relates a lot to the type of problem that we will be dealing with, in which the communication learned serves to improve performance on the task first and foremost. However, while the results were promising, the games or contexts used in these studies were limited.

More recently, research on the topic has continued with the use of neural networks, or more specifically forms of deep reinforcement learning. The goal of these approaches is to make the agents develop a language or communication protocol that enables them to play the cooperative game optimally (Foerster, Assael, de Freitas, & Whiteson, 2016; Lazaridou et al., 2017; Sukhbaatar, Szlam, & Fergus, 2016). In this work, the nature of the task is of great importance, as it determines the kind of feedback necessary to enable agents to learn. Common characteristics of the developed tasks are that they are fully cooperative and partially observable. Several variations of deep Q-learning enable the agents to indeed develop a useful language or communication protocol.

The advantage of such an approach is that the focus is on performance on the task itself, and not necessarily on creating perfect (human-like) communication, which is more relevant in collaborative tasks. Also, it is not necessary to gather a lot of data to train the agents, they simply have to play together many times. This, however, brings up a disadvantage as well; in order to properly learn how to communicate well in a (still relatively simple) task, agents will have to play the game thousands of times. When moving to the context of human-agent teaming, agents might sometimes have to train together with humans, who are not able to execute their task so many times in order to train the agent in communication. Also, in tasks that are more open and unpredictable, it is harder to optimize a communication policy for the task.

2.2.3 Relevant Methods for Human-Agent Teaming

As established before, one of the main aims of the current work is to create agents that can adapt their communication to human team members. Therefore, while using slot-filling systems or other rule-based methods might enable a good performance on a simple task, it is not fitting to the challenge of making agents more proactive and adaptive in their communication. Some kind of learning system is necessary. Since we are dealing with a very specific task and a clear collaborative context, the basis of the approach taken in this thesis is similar to the abovementioned methods that use collaborative games with a specific task for learning communication. Just like in those approaches, in this thesis agents will attempt to learn the right communication for coordination by getting feedback on the task, and not directly on the communication. The existing work with such methods has however never been applied to contexts with humans. This shows in the way their learning models have been built up; they are completely data driven models, that take many runs to learn proper behavior while there is little control over both the amount of messages sent as well as the quality of the messages.

For that reason, it was relevant to look at the existing methods for learning dialogue, which is actually human-agent. Most of the approaches are however supervised, where agents are exposed to large amounts of text. This is unfortunately also the case in the promising Memory Networks. This does not match well with the method of learning communication by getting feedback on a collaborative task. Methods that do match, are those that make use of reinforcement learning, as in MDP's or POMDP's. Using only those will however leave us with the same problem of needing many runs for learning, having little control over the quality of communication and making it impossible to train with humans. For that reason, it was chosen to attempt to combine the use of such reinforcement learning methods with rule-based methods.

2.3 Communication in Human-Agent Teaming

With knowledge about both human teams as well as the technical side of communication systems, it is relevant to look at research specifically into human-agent teaming. Quite a large and diverse body of research exists around this theme. Several experiments have for example been done on grounded or situated communication, to make sure that agents know what they are communicating about (Chai et al., 2014; Fang, Doering, & Chai, 2014; T. Williams, Acharya, Schreitter, & Scheutz, 2016). In the current work, the focus is, however, on specific communication patterns used as well as proactive communication, in which agents start communications without being prompted by a human.

2.3.1 Specific Communication Strategies

When looking at what and how to communicate in a joint task environment, there have been a few studies that test what kind of information has the largest effect on the improvement of team performance when communicated. Two examples clearly report a result, of which one is focused on agent-only teams (Harbers, Jonker, & Van Riemsdijk, 2012), while the other one also conducted experiments with humans (Li, Sun, & Miller, 2016). Both only look at quantitative performance measures and do not take into account more social, qualitative measures such as trust.

Both of the abovementioned studies test the effects of the communication of world knowledge or beliefs and that of intentions or goals. Both find that communicating goals or intentions is more effective than communicating beliefs or world knowledge in agent-only teams in increasing task performance. In the human experiment performed by Li et al. (Li et al., 2016), a similar result was found, although the increase in performance was small.

2.3.2 Proactive Communication

One study tried to actively create proactive (or anticipatory) communication strategies by adaptively optimizing the communication type (e.g. goals or beliefs) for different situations (Butchibabu, 2016). Proactive communication focuses mostly on implicit communication strategies, where agents proactively share necessary information. It can however also imply proactive explicit communication, in the case of for example proactively asking for help. The model created (a variation on an MDP) was able to increase task performance in agent-only as well as human-agent teams to a larger extent than any set strategy. However, the model did not take the context of the task into account, as it only looked at previous messages. Also, again the performance was only measured quantitatively by task performance.

More extensive ideas about how to enable proactive communication exist in the literature on mixed-initiative interaction, in which both agent as well as humans can take initiative to start communication (Allen, Guinn, & Horvtz, 1999). For example, there exists work on an algorithm that adaptively learns when to communicate what information based on participant roles, characteristics of the most recent utterance and dialogue history (Chu-Carroll, 2000), enabling the agent to base the decision on slightly more than just the previous message, making initiative in long periods of silence possible as well.

2.3.3 Relevance to Proactive Human-Agent Teaming Communication

Very little research exists on the learning of proactive communication in human-agent teaming. The work that comes closest to both the aims as well as the approaches of the current study is the work done by Butchibabu. In their work, agents also learn to communicate proactively using an MDP-based approach, while clearly focusing on human-agent teaming and not just agent-only communication by evaluating in an experiment with humans. The work presented here deviates from their approach by taking into account context factors outside of the actual communication, as inspired by the work on Social Practices (V. Dignum & Dignum, 2015). Also, while their model was learned offline on data gathered from human experiments, the focus here will be on online learning, such that agents are able to train with humans while still learning their own unique behavior.

2.4 Evaluation of Communication in Teaming

In order to study human-agent teaming, it is important to start understanding how the performance of communication between a human and a machine in the context of teaming can be evaluated. Different fields of research related to communication have used different methods and points of view to do this. Human-agent or agent-only team performance is mostly just evaluated on task performance. In the human teaming literature, however, more subjective and qualitative measures are used. The main approaches across disciplines will be discussed in the following section.

2.4.1 Task Performance

In many contexts, when the performance of task-oriented communication and generally of a team is evaluated, the most used and most important measure of evaluation is simply the improvement of the task performance (Butchibabu, 2016; Harbers et al., 2012; Li et al., 2016). Different task performance measures can be used depending on the task at hand, such as completion time or precision. However, using only task performance leaves out any details about why there might or might not be an improvement. It does not critically look at the quality of the communication, the way the human team members feel about the interaction or the actual understanding the system has of a situation. Therefore, it seems necessary to look into other evaluation measures.

2.4.2 Usability Evaluation

More qualitative evaluation of communication done by agents is mostly done in the field of interaction design and more specifically when dialogue systems are evaluated. The aim of such evaluations is to find the relation between certain interaction parameters and user quality judgements or usability (Dybkjær & Bernsen, 2001; Möller, Smeele, Boland, & Krebber, 2007). A term for these relations is Quality of Experience (Möller, Kühnel, Engelbrecht, Wechsung, & Weiss, 2009), or sometimes simply ‘quality’. It can be said that this ‘quality’ is a compromise between expectations about a system and actual perceived properties (Möller et al., 2007). Therefore, it is important to identify what users desire of a system when trying to evaluate their judgement of quality, to be able to place it in perspective. General aspects that are entailed by the term Quality of Experience include (i) interaction quality, (ii) efficiency-related aspects, (iii) usability, (iv) aesthetics, system personality and appeal, (v) utility and usefulness and (iv) acceptability (Möller et al., 2009).

Related more closely to the topic of usability in communication, factors that are directly relevant for quality in the evaluation of spoken dialogue systems can be identified as well: (i) output phrasing adequacy, (ii) feedback adequacy, (iii) adequacy of dialogue initiative, (iv) naturalness of the dialogue structure, (v) error handling adequacy and (vi) sufficiency of adaptation to user differences. These factors can be evaluated most effectively through user contact in the form of interviews or questionnaires (Dybkjær & Bernsen, 2001).

2.4.3 Communication Correctness and Efficiency

Related to the specific aspect of ‘output phrasing quality’ in usability, communication can also be evaluated purely on the quality of the utterances. Such methods are mostly used in unsupervised communication models; models that are not task-focused, such as chatbots meant for chit-chat, and can therefore not be evaluated with a task performance measure (Liu et al., 2017).

An example of such evaluation methods is to compare proposed utterances to ground-truth responses. By looking at word-overlap similarity metrics and word embedding metrics, the level of correctness of communication can be measured. However, such metrics have weak or no correlation to human judgement of communication (Liu et al., 2017).

Correctness can also be measured by having a human judge communications as correct afterwards, instead of comparing to a ground-truth (Griol, Hurtado, Segarra, & Sanchis, 2008). This can be combined with an efficiency measure by counting the number of communications and comparing that with task performance. Both these measures give insight into the quality of communication, but they do not give a full overview of the usefulness of it.

2.4.4 Trust Evaluation

It is well known that trust is an important factor in teaming activities that greatly affects task performance as well.

Trust in teamwork can be defined as the level of "willingness to accept vulnerability based upon positive expectations of the intentions or behavior of another", and allows team members to assume that possible conflicts or vulnerabilities will be resolved positively (De Jong, Dirks, & Gillespie, 2016). In terms of proactive communication specifically, trust is definitely influenced by people's expectations to be informed about aspects that relate to their responsibilities. Moreover, there are studies that specifically look at trust in human-agent teams, related to communication. It has for example been found that people have higher trust in systems that are transparent about their intents (Schaefer, Straub, Chen, Putney, & Evans, 2017). This might relate to the definition given above, since people can predict the outcome of a situation more easily if they know about the intentions of their team members.

However, while trust in team members even if they are computers is important, it is difficult to enable the development of long term trust in such situations. The development of trust requires repeated encounters and collaborations, which is not always possible when working with 'agents' (Riegelsberger, Sasse, & McCarthy, 2003). Research therefore resolves to look more at swift trust, as is done in the research on human virtual teams (Jarvenpaa & Leidner, 1999). It is then found that this kind of trust relates highly to quality of interaction, as it is also an important factor in the adoption of new technologies. It is most easily evaluated through questionnaires or interviews, where a Visual Analogue Scale is often used (Nasirian, Ahmadian, & Lee, 2017).

2.4.5 Relevant Metrics in Human-Agent Teaming

Looking at the different researched methods and metrics for evaluating dialogue, it is necessary to identify which are relevant for a human-agent teaming context. Task performance should be the number one evaluation metric. Since we are working in a safety-critical environment, it should be possible to guarantee a minimum level of performance and always strive for improvements in this area. However, since it is known that certain factors correlate with task performance, it is relevant to look into those as well, especially to gain more detailed insights on why task performance increases or decreases.

First of all, dialogue efficiency is relevant due to the nature of the task. In time-pressured tasks, all team members have their own responsibilities and it is not always beneficial to the task to communicate more. Therefore, it would be better if the system would not communicate too much in general; an efficient communication strategy is desirable.

In addition, trust correlates with team performance, as it improves the long term relationship that team members have with each other. Measuring trust is therefore useful for being able to reliably predict good task performance in the future.

Last, the general usability of the system is important too, as users will be working with the system over a longer period of time. As described above, aspects such as adequacy of initiative can be measured to understand whether the user is able to work with the system pleasantly. While these might also influence trust, they again give a more detailed view on why users might or might not trust the system.

These factors will therefore serve as the definition of team performance as a whole, and will be used in evaluation the developed proactively communicating agents later in an experiment with humans.

3. Learning Communication

As shown in the literature review in Chapter 2, there are many different ways of approaching the communication of an agent. There are also many different ways of approaching the learning of communication. None of the existing work, however, adapts the learning process to a team that includes human team members. The involvement of humans creates new requirements that have to be taken into account, that might be difficult to simulate in agents. This chapter discusses the approach taken in this thesis in an attempt to develop a method for making agents learn to communicate with human team members, supporting choices with literature.

3.1 Hybrid AI

It can be seen from the literature that creating a communicating agent is a difficult problem. Current solutions include the use of different types of technologies and algorithms. For this thesis, a deliberate choice was made for a hybrid approach that combines the advantages of two methods: BDI-agents and Machine Learning. There are a few reasons why this is a valid choice, especially within the defense domain. In such a safety-critical domain, leaving all behavior of an agent to machine learning is a risk. Machine Learning agents will always make mistakes, and in many domains this is to some extent acceptable, but in the defense domain much less so. Related to this, learning the correct behavior in a purely data-driven manner often takes time and requires large amounts of data. In unpredictable safety-critical environments, these amounts of data as well as the necessary time are mostly unavailable.

A purely rule-based agent, on the other hand, is often too rigid. Agents have to have the ability to adapt to environments and situations, as they might come across new situations that require behavior outside of the rules. To try to cope with this problem, relying only on rules often leads to extremely careful design which can lead to annoyance, especially in the case of communication. It follows that some aspects of teamwork (especially related to communication) are too fuzzy or too fluid to be expressed in rules, as we humans behave very intuitively on such aspects. It is exactly those aspects for which learning a probability distribution in a data-driven manner could be very beneficial. Finding out which aspects can be expressed in rules and which should be learned is an important challenge in getting an agent to learn proactive communication for teamwork. Below, a short description is given of both methods, including an explanation of how the two were combined.

3.1.1 BDI-Agents

The BDI (Belief-Desire-Intention) paradigm is a well-known model in the area of agent programming. Agents that are built using this model reason with beliefs, which are pieces of knowledge or observations about the environment which they believe to be true, desires or goals, which are states of the environment that the agent is aiming to achieve, and intentions, which are rules that relate to how those goals should be achieved (Rao & Georgeff, 1995). BDI-agents have been widely used in Multi-Agent Systems research (Deljoo, Gommans, Van Engers, & De Laat, 2017; Logan, 2015; Vergunst, 2011; White, Tate, & Rovatsos, 2017). Consequently, there are several Agent Programming Languages based on the paradigm, such as 2APL (Dastani, Mol, Tinnemeier, & Meyer, 2007), GOAL (Hindriks, 2009), Jason (Bordini, Hübner, & Wooldridge, 2007) and JACK (Busetta, Rönnquist, Hodgson, & Lucas, 1999).

The largest advantage of the BDI model, as well as agent programming in general, is that it focuses on the creation of autonomous decision making systems. It enables a system to decide to execute a certain plan based on the context at hand, where it is able to deal with dynamic environments and change plans when a plan fails. This makes the behavior of the agent in general understandable and interpretable in a way that is similar to humans.

However, implementing an intelligent agent is still quite a cumbersome task, even when using an Agent Programming Language. Part of that is caused by the limitations of the paradigm, such as the inability to deal with cost, preferences, time and resources, the inability to adopt a plan if more than one are applicable and understanding when to drop a goal if it cannot be reached (Logan, 2015, 2017). Especially in (human-agent) communication, where many actions are available and it is not always clear which one will give the desired result or how much the cost of an action will be, using merely BDI-agents will usually not give a good result.

3.1.2 Machine Learning

An area within the field of AI that can particularly deal with the limitations of BDI-agents, is Machine Learning. Machine Learning algorithms enable computers to learn, based on statistical regularities and patterns. There are three main types of Machine Learning:

- Supervised Learning: using labeled data, the algorithm tries to find a mapping between input and output to be able to make predictions about new data points.
- Unsupervised Learning: unlabeled data is used to make clusters or characterizations.
- Reinforcement Learning: an algorithm learns the optimal policy (series of actions) for a certain environment. Actions trigger rewards, enabling an agent to learn new behavior by estimating the expected reward for a given action.

Since these methods are probabilistic, they are especially good at dealing with costs of actions, levels of uncertainty and choosing between probable alternatives. However, usually large amounts of data (or in the case of Reinforcement Learning, many simulation runs) are necessary to make a Machine Learning algorithm perform well, because it needs to learn everything from scratch; it is hard to incorporate already known knowledge into ML systems. In order to learn communication, there should either be a large amount of conversational data available for the specific context, or agents should train in communicating thousands of times. The latter could be done in simulation of course, but if they should train with humans, it is impossible.

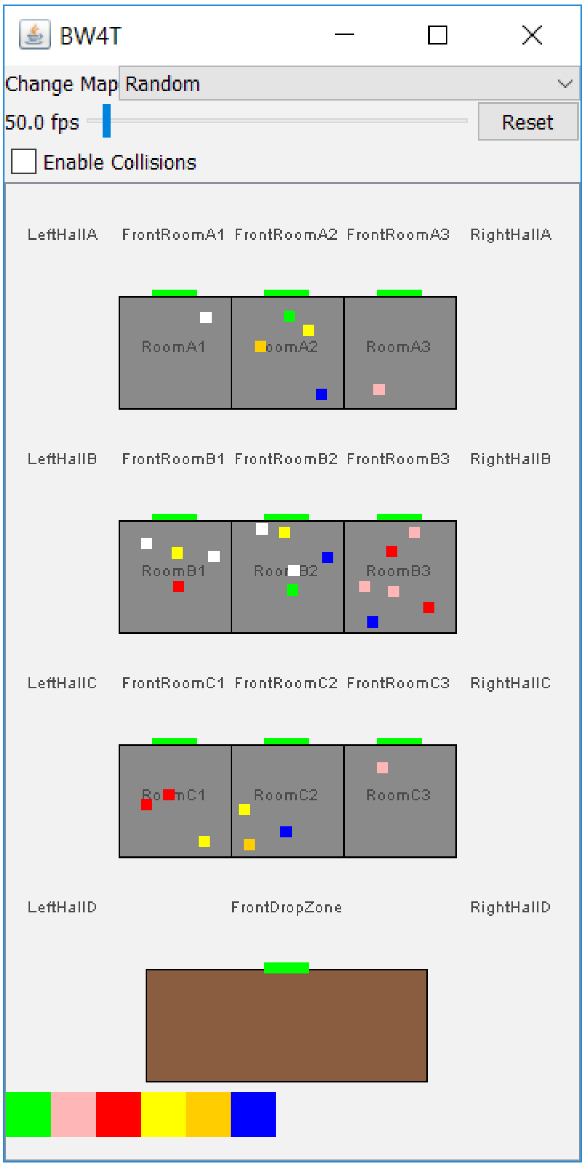
3.1.3 Combining BDI and ML for Learning Proactive Communication

Communicating proactively means that agents will have to know when to share which information, whether that is information about the world or about themselves, based on the current context they are in. In human-agent teaming, this can depend on the preferences and cognitive skills of the specific human the agent is collaborating with. Such factors are extremely hard to capture in rules, while a learning approach can help to capture the subtleties. However, there is no need to learn all communication from scratch, as we already know certain factors that influence whether information must be shared, such as whether it is relevant for the common task or whether it has already been communicated before. Using BDI-agents will ensure the ability to implement such prior known information as rules, while Machine Learning will help to learn the final subtleties of communicating the right piece of information at the right time.

To be more specific, as mentioned before, Reinforcement Learning will be used to enable agents to learn communication policies online. A supervised learning approach would need information about certain communication acts being correct or wrong in a certain context, which is very hard to capture by labeling data. Unsupervised learning cannot be applied in this context, since we do have a measure of when communications are good or not. Reinforcement Learning is able to distinguish more subtly the effect a communication has on the performance of a whole task by itself, without needing any hand-labeled data.

The two approaches were combined by letting a BDI-program determine the main behavior of the agent, with all actions, goal- and belief-management and dealing with received messages. A reinforcement learning algorithm was added on top to learn whether it is best to communicate or be silent in certain contexts. The definition of context, or the state representation, was determined mainly by specific beliefs defined in the BDI-program. A more detailed description of the agent and the different program parts will be given in later chapters.

A complication that might arise even in the combination of these two methods, is that it might still take quite some time for the agent to learn basic communication behavior. Having humans train with agents for a long time is not very feasible. Therefore, referring back to one of the sub research questions SQ5, agents will be trained in simulation first, after which they will continue training with humans in an experiment.



Change Map Random

50.0 fps

□ Enable Collisions

Figure 1. Blocks World 4 Teams: the basic scenario

3.2 An Environment for Learning Proactive Communication

To create a proactive communication system for human-agent teaming, it is necessary to have an environment in which the system can operate in such a way that it can be easily tested. In particular, a testbed is needed that allows for a combined human-agent team to do a task in such a way that it represents real-life use case scenarios to a satisfactory extent.

Several testbeds have been developed for multi-agent systems, but only very few that allow for human participation as well. There are basically two which are described in the literature: Gamebots 3D (Adobbati et al., 2001) and Blocks World 4 Teams (M. Johnson et al., 2009). Both of these testbeds offer the possibility of playing in human-only, human-agent and agent-only team configurations. Team members can work on the same task, where a team member or 'bot' can either be sent around to do its task by a human through a graphical user interface, or by an agent program.

The difference between the two is that while Gamebots is a full 3D environment based on a commercial game engine, BW4T is a relatively simple 2D Blocks World environment. Gamebots offers more in terms of functionality, where different types of games can be played and teams can compete or collaborate. BW4T, on the other hand, currently only offers one simple task on which one team works. The simplicity makes the environment more accessible for changes made by a researcher than Gamebots, even though Gamebots does offer several extra features for the collection of research data. Also, BW4T was designed with the specific aim of creating a research tool for human-agent coordination and communication in teaming, whereas Gamebots is more of a general environment.

Gamebots has many desirable features, but the simplicity and accessibility makes BW4T a better tool as a starting point for an environment for the current study. It was also used in several studies of which the conclusions and approaches relate closely to our research questions (Butchibabu, 2016; Harbers et al., 2012; Li et al., 2016).

3.2.1 BW4T - Detailed Description

The base of the BW4T testbed is a simple Blocks World environment, consisting of rooms containing colored blocks which are connected by corridors (Figure 1). Next to the general block-containing rooms there is also a special drop zone. The task that is programmed in the environment is represented by colored blocks in the lower left corner of the screen. The agents in the environment have to search the rooms for blocks in the right color, and deliver them to the drop zone in the same order as the task describes.

Agents can see the environment and the locations of the rooms, but they can only see what blocks are in a room when they are in that specific room. This means that in order to find the right blocks, they have to explore and go through different rooms. Agents can only see each other when they are near each other, meaning that they usually do not know where their team members are. They can however see when another agent is in a certain room from the color of the door: the door is green when a room is free, and red when a room is taken. In the graphical user interface for humans directing bots, this is the same (Figure 2).

Essentially, agents could perform the task completely on their own. However, when working in a team, it is easy to see that the task could be achieved much quicker when team members actively collaborate. The creators of BW4T call this soft interdependence or coactivity (M. Johnson et al., 2011). To enable such collaboration, it is necessary for agents to communicate.

Tasks with different levels of complexity can be defined, where the number of different colors and the importance of order can be changed. Also, handicaps such as color blindness can be added to the bots to make it more difficult for them to achieve their task. One limiting property that all bots already possess is a battery that slowly uses power. By adding a charging zone in the environment, the bots can charge their battery. Using the charging mechanism makes completing the task more complex as well.

3.2.2 Relating BW4T to Real Life Scenarios

Real life human-agent teaming scenarios contain elements of which some are more present than others in the classical BW4T scenario. Some elements of real-life human-agent teaming scenarios are the following:

1. Independence of Task Execution: Team members have their own tasks that they should be able to mostly execute independently. This is the case in many teaming scenarios, of which one is the Search and Rescue use case.

2. Shared Situational Awareness: It is helpful if team members maintain a shared situational awareness, meaning they have a clear overview of the environment and task and know to what extent their team members have this overview as well.

3. Possibility of Improvement by Collaboration: In basically any teaming scenario, and maybe even more so in a human-agent teaming scenario, it must be possible to achieve a better result when collaborating than the team members can achieve

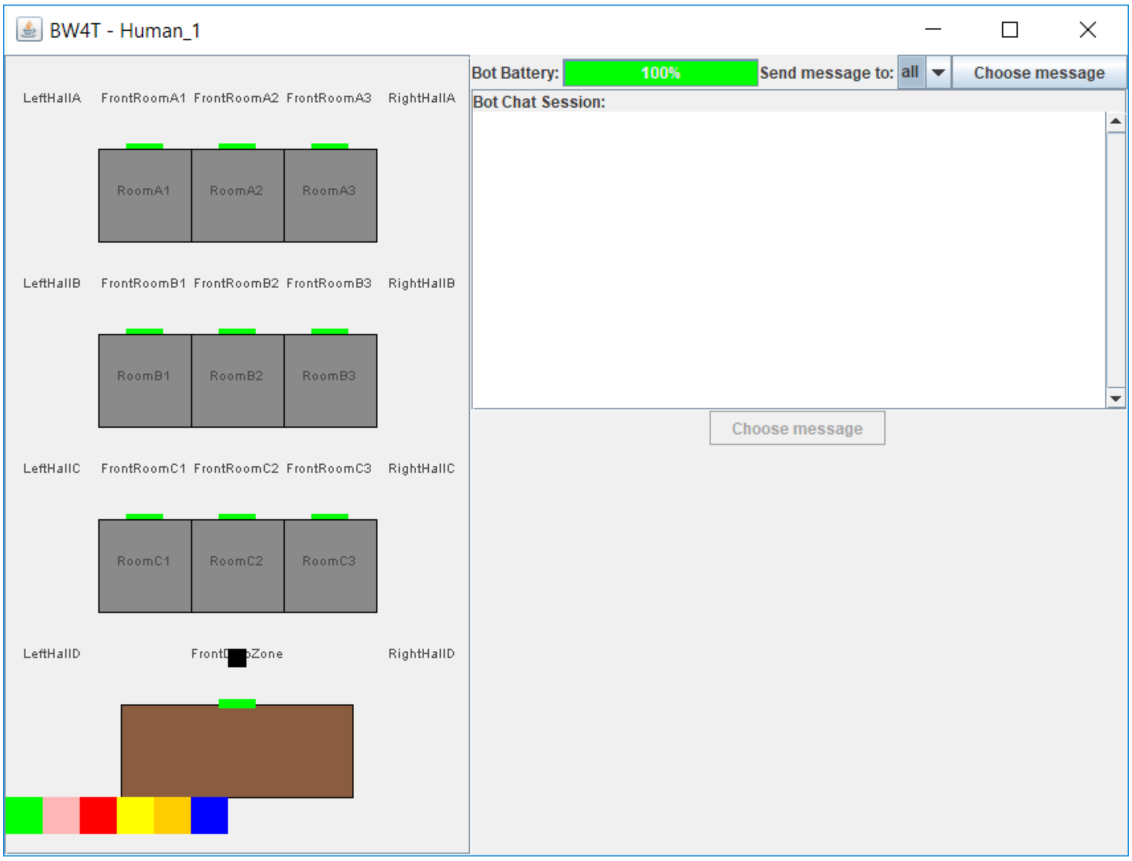


Figure 2. BW4T: The human GUI

on their own. This can be caused for example by the different capabilities and knowledge of the team members. Dividing tasks, taking up roles and asking for help all are factors that enable this.

4. Unpredictability of Environment: In any real life task, unexpected events might or might not occur in the environment.

The first two of those factors, (1) independence of task execution and (2) shared situational awareness, are definitely present in BW4T. In BW4T, agents can finish the task independently, while it is beneficial to keep a shared situational awareness of the blocks that have been found in the different rooms. The third factor, (3) possibility of improvement by collaboration, is present to a certain extent. The fact that the task is executed by more team members enables a higher performance, and dividing tasks can increase the advantage. However, in the current scenario there is no clear difference in capabilities between different team members, meaning that while it is beneficial to collaborate because of the possible difference in knowledge, it is not necessary. The final factor, (4) unpredictability of environment, is not currently present in BW4T.

The Final BW4T Scenario

The final BW4T scenario that was used in this thesis is the setup with nine rooms and a Dropzone (as depicted in Figure 1). In smaller setups it would not always be beneficial to communicate or collaborate, while a larger setup would make the task unnecessarily more complex. Also, the presence of a range of possible starting points as well as the possibility of large distances between the agents ensures that trusting your team member is not trivial.

It was decided to not use the battery function or the different 'handicaps' that can be added to agents. While using these functionalities would increase the similarity between the BW4T task certain real-life scenarios (the handicaps for example relate to factor (3)), it would also make it harder to complete the task. Since the agents would only learn communication and not actions, using those functions would make it very likely for fatal errors to occur, making it difficult to evaluate team performance over many runs. In the future it will definitely be interesting to look at the effect of adding functionalities like these. For similar reasons, the task currently was not changed to add factor (4).

In first evaluations of the learning agents, the 'Rainbow' task was used, which is a task consisting of blocks in all colors of the rainbow, in that order, with a fixed distribution of blocks over the rooms. To enable a fair evaluation of performance later on, a random task sequence and distribution of the blocks was used, with a set length for the sequence of six blocks. The general programmed behavior of the agents can be found in Appendix C.

3.3 Learning in GOAL

Since we are working with a combination of a BDI-agent and a form of Machine Learning, it would be useful to look into existing combinations of those. Most work has been done on using some form of machine learning to make an agent learn a preference over actions which have the same preconditions but cannot be executed simultaneously (Airiau, Padgham, & Sardina, n.d.; Bădică, Bădică, Ganzha, Ivanović, & Paprzycki, 2016; Broekens, Hindriks, & Wiggers, 2012; Deljoo et al., 2017; Nguyen & Wobcke, 2006; Singh & Hindriks, 2013; Singh, Sardina, Padgham, & James, 2011). The major goal of this work is to tackle one of the most important problems that BDI-agents have: conflicting goals or actions might appear, making an agent unable to perform any action. Within the GOAL Agent Programming Language, which is compatible with the BW4T environment, it was attempted to solve this problem by having an agent learn preferences over actions, given the current set of beliefs and goals of the agent (Singh & Hindriks, 2013). Using the beliefs and goals of the agent as an input is a way of acknowledging that the agent does not always have full knowledge about the environment. This will be beneficial when moving towards a real world scenario. Because of this, the existing learning module of the GOAL language was used in this thesis.

3.3.1 GOAL Agent Programming Language

The Agent Programming Language GOAL uses rules to describe the behavior of an agent in relation to its mental state (beliefs and goals). The mental state is represented in Prolog, which can be updated through the use of rules in the GOAL code. Rules consist of a condition that evaluates the mental state of the agent and executes some action if the condition holds.

Rules can be specified in modules and a module can be executed as an action within another rule. Within GOAL, a certain order of executing rules can be specified for each separate module. One of those specifications is 'adaptive', making the agent adaptively choose the best suitable action from all applicable actions in the module, using Reinforcement Learning.

3.3.2 Q-Learning

The form of Reinforcement Learning currently implemented as the learning algorithm for GOAL is Q-learning. The basic idea of Q-learning is that at each timestep *t*, the agent is in state *s*, and chooses an action a to transition to the next state *s'*. A reward *R* is then returned and used in the Q-function to determine the expected return for performing action *a* in state *s*.

$$Q^{\pi}(s,a) \leftarrow Q^{\pi}(s,a) + \alpha\left[R(s,a) + \gamma\, max_{a'}(Q(s',a')) - Q^{\pi}(s,a)\right]$$

In this function, α is the learning rate that determines how much influence the new reward has on the existing Q-value. To learn the optimal Q-values, agents try out different actions and keep a table of Q-values for every state action pair that is constantly updated. In order to learn an optimal policy, agents must find a balance between exploration and exploitation; exploring helps them to find new optimal actions, while exploiting current knowledge helps to converge to a final policy

Since Q-learning keeps representations for all possible state-action pairs, it does not scale well to problems with large state spaces. For example using Neural Networks as done in Deep Q-learning enables agents to generalize their learned behavior to new states, ultimately leading to better performance. In the current study, however, simple Q-learning was used, to be able to find out if learning of communication is possible with such a basic Reinforcement Learning algorithm.

3.4 Levels of Complexity in Agent and Communication Strategy

To approach the development of communication learning agents in a systematic way, different levels of complexity of the BDI-agents as well as of the communication learning algorithm were defined. These definitions have been used as a guideline for behaviors agents should have in order to enable the learning of communication. Some of those will be implemented throughout the rest of the thesis. This overview serves as a reference of how far the work done reaches, and what steps still have to be taken in order to enable complete proactive communication. The defined levels are based on different possible teaming functionalities that agents could have in BW4T, but they are applicable to general teaming scenarios as well.

3.4.1 Levels of Agent Bevavior Complexity

The defined levels of agent behavior complexity assume a rule-based BDI-agent. For all named functionalities it must therefore be possible to implement them in rules. The levels deal with the receiving side of the communication; they define how agents make use of received communication done by their team members.

A1	Agents reason only with beliefs about the environment (e.g. rooms and blocks) and their own goals. They know of the existence of their team members (by ID), but are agnostic to their beliefs, goals, plans and actions. They are able to receive messages and use the information in those messages to update their knowledge, but they only use the explicitly mentioned information and only use it to update knowledge about the environment. They are not able to reason about implicit implications of this information. Agents can send messages, but without clear goal or knowledge about what their team members might do with the received message. Agent behavior is the same as it would be if they were performing the task on their own; they do not take into consideration what the other might do in their actions.
A2	Agents reason with beliefs about the environment, their own goals and goals of the other agents that were communicated. This means they automatically decide to not pursue a goal if a team member already pursues the same goal, and move on to what might be necessary after that goal is achieved. This does not mean they can understand actions that might implicitly follow from this goal; they assume the goal state of the other agent is achieved and should therefore be skipped.
A3	Agents reason with beliefs about the environment, their own goals and goals of the other agents that were communicated. This means they can decide to not pursue a goal if a team member already pursues the same goal and anticipate on what might be necessary after that goal is achieved. They might as well decide to overwrite the commitment of their team members if that is considered more beneficial to the task. They can understand actions that might implicitly follow from this goal and make an estimate of how soon the goal state of the other agent will be achieved, to decide if it can be skipped or waited for.

3.4.2 Levels of Communication Learning Complexity

The levels of communication learning complexity were defined as instances of a reinforcement learning problem that decides about a communicative act. They show the action possibilities Am that the agent has to choose from and the possibly communicated information. The different types of to-be-communicated information is based on the distinction between reactive-implicit, deliberative-implicit and explicit information.

C1	During every cycle *c*, for all percepts *block(Block, Color, Room)*, evaluate state *s* and determine action with highest expected reward *a* from A_m = {*communicate, silence*}.	Decision when to be communicated information is conceived

C2	During every cycle *c*, for all beliefs *block(Block, Color, Room)*, evaluate state *s* and determine action with highest expected reward *a* from A_m = {*communicate, silence*}.	Decision when to be communicated information exists
C3	During every cycle c, for all beliefs φ, evaluate state *s* and determine action with highest expected reward *a* from A_m = {*communicate, silence*}.	Decision when to be communicated information exists
C4	During every cycle *c*, for all goals γ for which holds that *adopt*(γ) in *c* and not γ in *c-1*, evaluate state *s* and determine action with highest expected reward *a* from A_m = {*communicate, silence*}.	Decision when to be communicated information is conceived
C5	During every cycle *c*, for all goals γ, evaluate state *s* and determine action with highest expected reward *a* from A_m = {*communicate, silence*}.	Decision when to be communicated information exists
C6	During every cycle *c*, for all goals γ and beliefs φ for which holds that *adopt*(γ) and *insert*(φ) in *c* and not γ and φ in *c-1*, evaluate state *s* and determine action with highest expected reward *a* from A_m = {*communicate, silence*}.	Decision when to be communicated information is conceived
C7	During every cycle *c*, for all goals γ and beliefs φ, evaluate state *s* and determine action with highest expected reward *a* from A_m = {*communicate, silence*}.	Decision when to be communicated information exists
C8	During every cycle *c*, for all goals γ and beliefs φ, evaluate state *s* and determine action with highest expected reward *a* from A_m = {*communicate, silence*, $a_{exp1} \dots a_{expn}$}.	Decision when to be communicated information exists

4. Learning to Communicate Beliefs

Looking at the different levels of complexity of communication and agent as defined in the previous chapter, a logical first step towards achieving an agent that learns when to communicate what information in a human-agent teaming context is to try to make the agent learn how to communicate beliefs. At the lowest level of communication complexity (C1), this is reduced even further to a specific type of beliefs: blocks that were perceived by the agent and subsequently stored in the belief base. This way, the agent only has to deal with one specific type of information to start with. In other tasks or environments this can be replaced by a specific type of static information about the state of the environment. Since it is information that could essentially be perceived by every agent by itself (assuming all agents have the capability to observe blocks, as is the case in the chosen scenario), communications about blocks can easily be processed by an agent of low complexity. The agent will simply have to add a belief about the communicated block to its own belief base, for which no complex reasoning is necessary. An agent that would be able to do that was described at agent complexity level A1. At the same time, the communication of observed information is a relevant type of communication in teaming use cases as it increases the shared situational awareness of the team members, while also easily causing an information overload when communicated too often. This makes it a suitable starting point for learning a balance between communicating and not communicating at the right moment as well.

In the current chapter, a process working towards agents that are able to learn in what situation they should communicate about blocks they perceived, is described. First, a description of challenges within the process is included, especially focusing on adapting what was originally the learning action selection to the specific case of learning communication in the BW4T environment. After that, different levels of implementation are discussed as well as results from experimental runs of agents in simulation, showing the learning process of those agents.

4.1 Adapting Learning to Communication

4.1.1 Problem Description

As described before, the communication learning will be based on the existing learning module for GOAL. This learning module, however, was made and tested only in a single-agent context . Additionally, the purpose of the learning module was for the agent to learn preferences over actions, in order to optimize the action selection process. The current context presents us with a multi-agent context, and while learning which communicative act to perform might be relatively comparable to learning which action to perform, there are some important differences. Consequently, the use of the GOAL learning module had to be adapted to fit the purpose and context. In order to do so, an attempt was made to write a detailed description of the challenges that are part of this adaptation.

In principle, the problem can be described as a classical Multi-Agent Reinforcement Learning (MARL) problem; several cooperative agents aim to learn how to solve a task collaboratively as quickly as possible, while receiving feedback from the environment (Buşoniu, Babuška, & De Schutter, 2008). However, due to several complications it differs from the classical problem. Research has been done on creating algorithms to work with these complications, but they are usually not dealt with in combination. The main problems are the following:

- **Diverse Agents:** since we talk about human-agent teaming, the agents in our multi-agent system will not all be the same internally. Usually MARL algorithms assume that all agents are the same (Panait & Luke, 2005). In our case, when we talk about Shared Situational Awareness, an agent can never assume with certainty that the other team member has certain knowledge or observations, due to possible differences in capabilities. This might require other, more complex communication strategies than in a case with agents that are all the same.

A possible way to solve this problem is by simply letting the agents learn only with feedback from the environment, without knowing anything about their team members. While this creates other problems (such as a large state space), it is possible to learn policies that way (Buşoniu et al., 2008).

- **Continuous or Large State Space:** in many real-life contexts, the state-action space to be used for learning is continuous. There is no set amount of states that one can be in, and new, unknown states will appear continuously. While this can be constrained for research purposes by using the BW4T environment, this environment still generates a large amount of states, making it hard to learn any communication policy using look-up tables as in regular Q-learning with the state representation of beliefs and goals. In particular, learning will take a long time and many runs, which is not desirable when an agent is to (partly) learn from interaction with a human. There are basically two solutions to this problem: the state representation should be more abstract or higher level, or the learning algorithm should be able to generalize learned preferences to new, similar situations, as attempted in (Mnih et al., 2015). While this is a well-known problem in Machine Learning, the challenge is to find the right level of abstraction to enable learning.

- **Presence of Hierarchies or Structure:** in the case of communication, actions usually don't come in isolation; they are generally part of a sequence or structure, creating a dialogue. While this can also be the case in tasks and procedures that have nothing to do with communication, it is particularly important in communication. Reinforcement learning is made to learn sequences of state transitions. However, in our context, there is not one sequence that runs from start to end. There might be several short dialogues in a full run, appearing in a different order every time. Within such short dialogues, however, the order of actions is often similar, although changes might occur. This asks for a certain hierarchy in learning; in a certain situation, agents should know which dialogue to start and try to stick to, but when there are deviations in the regular dialogue pattern, the agent should know how to respond to these on the atomic action level.

Existing solutions to this are the creation of very high level actions, or the use of hierarchical reinforcement learning, where the algorithm is able to learn compositions of actions as optimal behavior (Barto & Mahadevan, 2003).

- **Delayed, unreliable and fuzzy rewards:** communication often does not have a directly observable and real-time effect on the environment or task. While reinforcement learning is designed to deal with delayed rewards, as this problem is not unique to communication, it still assumes that state changes happen due to actions done. This direct causal relationship does not always hold when moving from one decision about communication to the next. Feedback on communication might be very implicit or only provided after a very long time, making it hard or near impossible to translate this to a reward for a specific individual communicative act. Rewards that are easy to interpret, on the other hand, are sometimes so simple that it would be easier and more efficient to capture them in rules.

In order to adapt the learning for the most basic types of communication, the most urgent challenges to tackle are those of the large state space and the delayed and unreliable rewards. Without at least approaching a solution to these challenges, learning would not be possible at all. The other two challenges, the presence of diverse agents and hierarchies, become relevant when the communication becomes more complex. For example, when the agent will try to actively adapt its communication strategy to the state of a human team member, it is relevant for the agent to understand that the human does not have perfect memory and attention, and solve that problem by communicating in dialogue rather than monologue. In the first simple example we will however not consider such forms of social intelligence and communication.

First Attempt at Learning

The first attempt at adapting the learning for communication implemented communication complexity C1 and agent complexity A1. This means that every time an agent would encounter a block it had not seen before, it would decide whether it was worth it to communicate the information, based on the expected reward for the action.

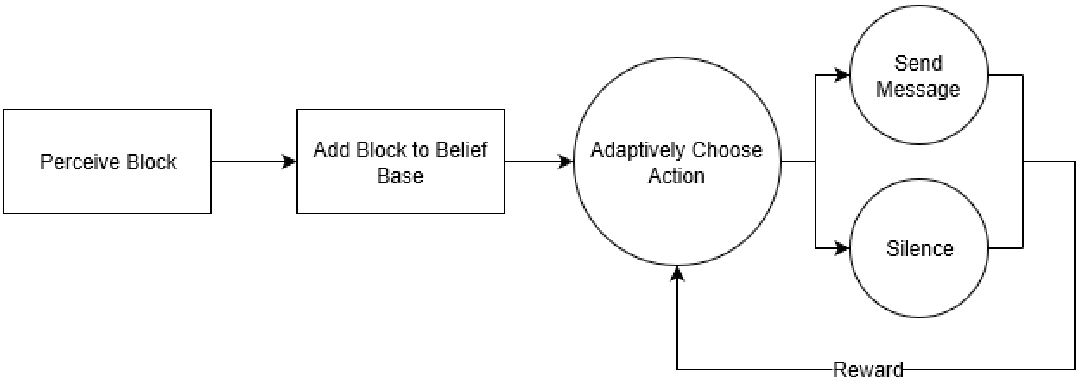


Figure 3. C1A1 Deciding about communication when perceiving a block

After choosing an action, the reward for that particular action would be evaluated and added to the chosen action. The rewards implemented in this first attempt were twofold:

- **Cost of Communication**: a cost was added for every sent message

- **Time Reward**: after finishing the sequence, a reward would be calculated based on the time it took to finish the sequence, where a longer time meant a smaller reward. With r_{max} being the maximum reward and *t* the time it took to finish the sequence in seconds or in simulation steps, this reward was calculated as follows:

$$R = r_{max} - t * 20$$

The height of the cost and the value of r_{max} was determined by looking at how often the adaptive module was run, to make sure that an average to slow completion time would yield a cumulative reward of zero. As a basic communication cost, a value of -1 was chosen.

However, after evaluating the accumulated rewards after every action, it seemed that the time reward was never added to the action. This was due to the delay of the reward: the last decision about communication is made when the last block is perceived in a room, while no decision about communication is made at the moment the sequence is finished (which is the moment at which the reward is calculated). It follows that with the current Q-learning algorithm, if a time reward is to be used, the learning module should run at the moment the time reward is triggered.

A solution to this problem is to move to a higher level of communication complexity, namely C2, in which a communication decision is made at every cycle *c* for every belief φ about a block that exists in the belief base. This way, a reward will be received for some action chosen, which can then be propagated to previous actions. Also, this gives the agent the opportunity to also learn about the timing of a communicative act, as it might learn to not communicate a certain piece of information until the context requires otherwise.

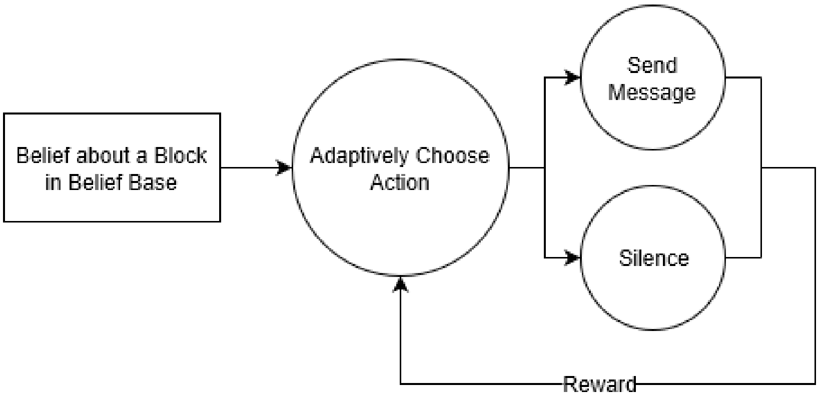


Figure 4. C2A1 Deciding about communication when having a belief about a block

4.2 Leveraging Rules and Learning for Blocks Communication

Adapting the learning, or in this case rather the complexity of the communication strategy, helped to use delayed rewards. In proactive communication, it will generally be important to evaluate whether information should be communicated or not every timestep, to make sure any feedback from the environment can be taken into account. However, in the new scenario, a decision about communication was made so often, that it became clear that the state space grew very quickly. In order to decrease the size of the state space, it was chosen to make use of the fact that the agent was a hybrid agent already. Since the states are built up of beliefs and goals, the size of the state space can be controlled by controlling which beliefs and goals actually end up in the state representation. Simply cutting down the amount of beliefs and goals, however, would eliminate much of the information that the agent could use to make a decision about communication.

A way to ensure a small state representation that still contains most of the important information, is to use rules in the BDI program to already incorporate some reasoning steps or domain knowledge before the learning process is started. For example, we might say that in order to decide whether we want to communicate information about a block, it is relevant to consider only two features: whether the information has been communicated before and whether the block is part of the task currently at hand. Any other information might be considered irrelevant for determining the right communication decision. This translates to two general factors that should be considered when deciding about communication in any context: (a) redundancy and (b) relevance to the task.

Transforming the above into beliefs fitting the BW4T environment and GOAL programming language could look as follows:

```
communicatedThis(boolean)
inTask(int/boolean)
```

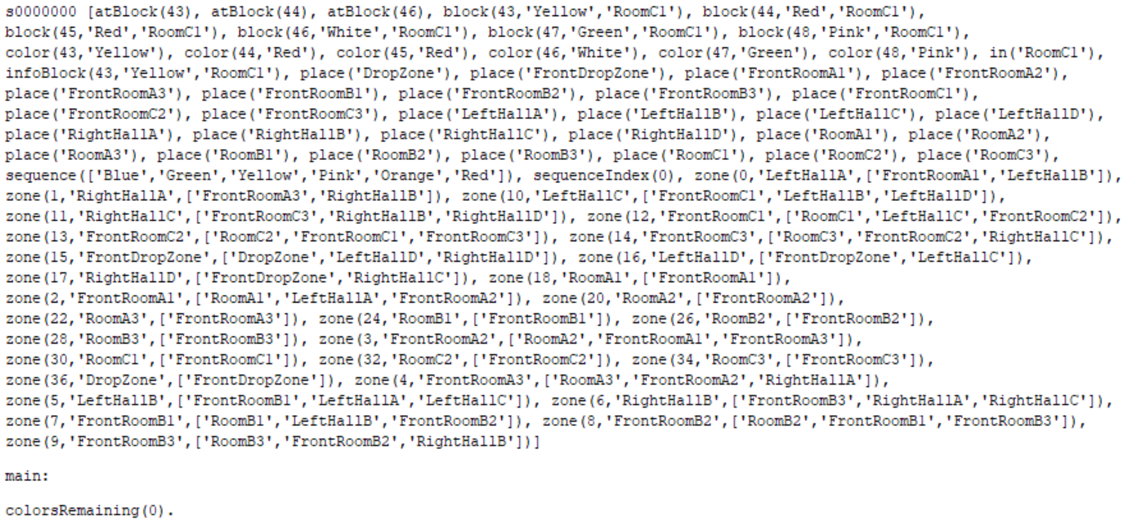
Here ‘communicatedThis’ becomes true if information has been communicated before. The belief ‘inTask’ checks whether the color of the block at hand occurs in the task sequence, and if this is true, it returns how many blocks in the sequence this color is away from the color the agents are currently looking for. These beliefs can easily be determined and controlled by the following knowledge rules:

```
communicatedThis(Block):- infoBlock(Block,Color,Room), communicated(Block,Color,Room).

inTask(Index):- infoBlock(Block, Color, Room), sequence(L), sequenceIndex(I), member(Color, L), nth0(Loc, L, Color), Loc>=I, Index is Loc-I.
```

In these rules, the ‘infoBlock’ belief stores a piece of information about a block that might be communicated, ‘communicated’ is a global belief that stores all previously communicated blocks, ‘sequence’ represents the task sequence and ‘sequenceIndex’ stores the color that the agents are currently looking for.

To get a view of the effect of using only these two specific, expert knowledge determined beliefs as state representation, we can look at the difference with the original state representation, as displayed in Figure 5. Using such a state representation counted up to a state space of about a thousand in the standard BW4T scenario after a hundred runs.



```
s0000000 [atBlock(43), atBlock(44), atBlock(46), block(43,'Yellow','RoomC1'), block(44,'Red','RoomC1'),
block(45,'Red','RoomC1'), block(46,'White','RoomC1'), block(47,'Green','RoomC1'), block(48,'Pink','RoomC1'),
color(43,'Yellow'), color(44,'Red'), color(45,'Red'), color(46,'White'), color(47,'Green'), color(48,'Pink'), in('RoomC1'),
infoBlock(43,'Yellow','RoomC1'), place('DropZone'), place('FrontDropZone'), place('FrontRoomA1'), place('FrontRoomA2'),
place('FrontRoomA3'), place('FrontRoomB1'), place('FrontRoomB2'), place('FrontRoomB3'), place('FrontRoomC1'),
place('FrontRoomC2'), place('FrontRoomC3'), place('LeftHallA'), place('LeftHallB'), place('LeftHallC'), place('LeftHallD'),
place('RightHallA'), place('RightHallB'), place('RightHallC'), place('RightHallD'), place('RoomA1'), place('RoomA2'),
place('RoomA3'), place('RoomB1'), place('RoomB2'), place('RoomB3'), place('RoomC1'), place('RoomC2'), place('RoomC3'),
sequence(['Blue','Green','Yellow','Pink','Orange','Red']), sequenceIndex(0), zone(0,'LeftHallA',['FrontRoomA1','LeftHallB']),
zone(1,'RightHallA',['FrontRoomA3','RightHallB']), zone(10,'LeftHallC',['FrontRoomC1','LeftHallB','LeftHallD']),
zone(11,'RightHallC',['FrontRoomC3','RightHallB','RightHallD']), zone(12,'FrontRoomC1',['RoomC1','LeftHallC','FrontRoomC2']),
zone(13,'FrontRoomC2',['RoomC2','FrontRoomC1','FrontRoomC3']), zone(14,'FrontRoomC3',['RoomC3','FrontRoomC2','RightHallC']),
zone(15,'FrontDropZone',['DropZone','LeftHallD','RightHallD']), zone(16,'LeftHallD',['FrontDropZone','LeftHallC']),
zone(17,'RightHallD',['FrontDropZone','RightHallC']), zone(18,'RoomA1',['FrontRoomA1']),
zone(2,'FrontRoomA1',['RoomA1','LeftHallA','FrontRoomA2']), zone(20,'RoomA2',['FrontRoomA2']),
zone(22,'RoomA3',['FrontRoomA3']), zone(24,'RoomB1',['FrontRoomB1']), zone(26,'RoomB2',['FrontRoomB2']),
zone(28,'RoomB3',['FrontRoomB3']), zone(3,'FrontRoomA2',['RoomA2','FrontRoomA1','FrontRoomA3']),
zone(30,'RoomC1',['FrontRoomC1']), zone(32,'RoomC2',['FrontRoomC2']), zone(34,'RoomC3',['FrontRoomC3']),
zone(36,'DropZone',['FrontDropZone']), zone(4,'FrontRoomA3',['RoomA3','FrontRoomA2','RightHallA']),
zone(5,'LeftHallB',['FrontRoomB1','LeftHallA','LeftHallC']), zone(6,'RightHallB',['FrontRoomB3','RightHallA','RightHallC']),
zone(7,'FrontRoomB1',['RoomB1','LeftHallB','FrontRoomB2']), zone(8,'FrontRoomB2',['RoomB2','FrontRoomB1','FrontRoomB3']),
zone(9,'FrontRoomB3',['RoomB3','FrontRoomB2','RightHallB'])]

main:

colorsRemaining(0).
```

Figure 5. Original state representation with all beliefs and goals

The new filtered state representation, on the other hand, might look as follows:

```
s0000000 [communicatedBlock(false), isInTask(1)]
```

Using this state representation creates a state space of fourteen. Also, with such a small state space it is possible to easily evaluate the conditions or 'rules' that the agent learns.

4.2.1 Evaluating the Filtered State Representation

To see what kind of effect the use of knowledge rules to filter beliefs for the state representation has on the learning process, some small experiments were run. The same scenario was used, namely C2 with A1, using the standard BW4T Rainbow map. The team playing consisted of two identical agents. For both state representations mentioned above, the game was run a hundred times with a message cost and time reward, with r_{max} being 1200.

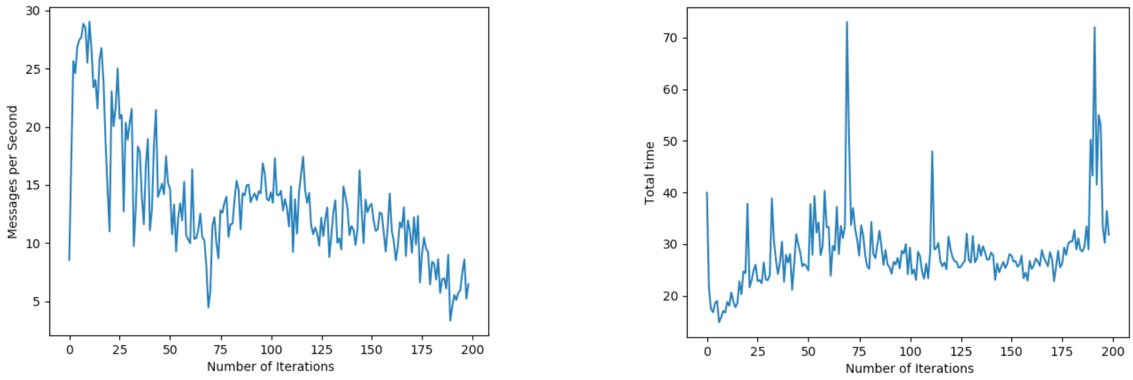


Figure 6. Original state representation

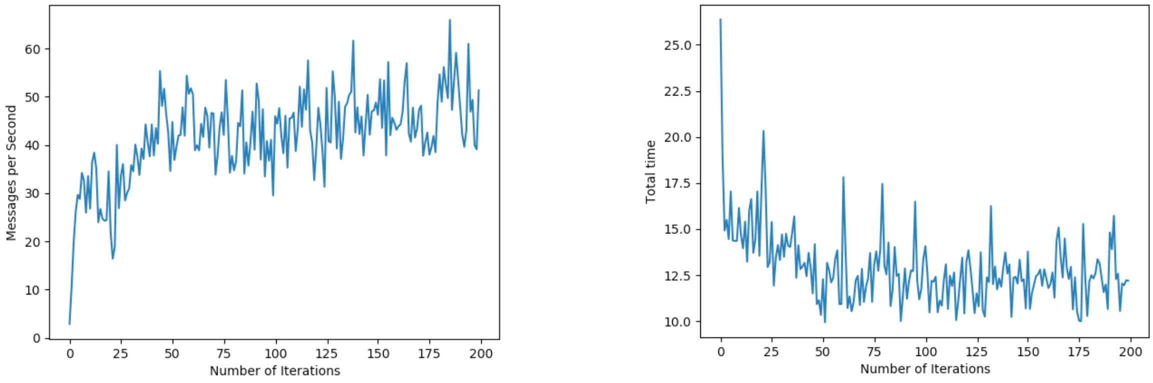


Figure 7. Filtered state representation

Both state representations show a learning curve of some sort. In both cases, the number of messages per second is relatively low in the first few runs, which can be explained by the fact that the message cost is already accumulating during one run, effectively teaching the agents that communication is bad. At the end of a game, however, the time reward is supposed to tell them that when they communicate, sometimes the sequence can be finished much faster. In the filtered state reward this effect is clearly visible, as the agents slowly start communicating more while the time they take to finish the sequence decreases. Also, the largest effect is achieved within about 50 runs, which is a very small number that might even be feasible for training with humans.

With the original state representation, however, it is much harder to explain the learning process. The number of messages first increases, but quickly starts a steady decline. Simultaneously, the total time it takes to finish the sequence seems to increase rather than decrease. In general, the total time is much higher than with the filtered state space.

When qualitatively looking at the Q-values for the filtered state it can clearly be seen that in general the agents learn that they should not communicate information that has already been communicated before. They also learn that usually, if something has not been communicated yet, it is smart to do so now. No clear behavior is learned on the basis of whether a color exists in the task however.

In conclusion, we can say that filtering the belief base with the use of knowledge rules helps to learn reasonable behavior quickly. However, it should be noted that the amount of messages sent is still very high. Even though the simulation was slightly sped up, 40 to 50 messages per second does not seem necessary. This might be due to a high r_{max} which rewards communication, but a high exploration rate probably contributes as well. When playing in agent-only teams this is not a big problem, but humans are prone to experiencing 'information overload' when they receive too much information. Considering that the task used consists of 6 blocks, it seems reasonable that the total amount of messages communicated should be somewhere between 6 and 18, considering that there might be more blocks of the same color that can be communicated. Last, it is noticeable but to be expected that no clear behavior is learned related to the 'inTask' belief, as with the perfect memory of the agents, it does not matter much how far away in the task a certain color is. When playing with humans, however, this might make a difference, so it could be interesting to see if the agent is able to learn this from playing with a human team member.

4.3 The Influence of Exploration and Rewards

One of the main problems that the belief-communicating agent has, is that it still sends out an enormous amount of messages. While this is not a big problem for an agent playing in simulation, it will not work when actually teaming up with (a) human(s), as the large amount of messages might confuse people and cause an information overload. By tweaking the reward and exploration rate of the Q-learning, it was attempted to lower the amount of messages to a more realistic number, while maintaining the 'rules' that the agent was able to learn.

4.3.1 Quantitative Analysis

Since it had to be made less profitable to communicate, either a lower r_{max} or a higher cost of communication could be tested. A choice was made to explore the effects of a different values for r_{max}. Similarly, a large exploration rate would cause agents to choose to communicate information more than once, which is undesirable, so this was lowered as well.

Experimental series of 200 runs per condition were evaluated, using values of 500 and 1200 for r_{max} with time in seconds, and using a value of 1200 and 2500 for r_{max} using deliberation steps. Next to that, four different exploration rates were used: 0.1, 0.01, 0.001 and 0.1 with a decay of 0.05. The BW4T environment was sped up, to allow for a quick evaluation of the best performing set of parameters. To improve readability on the time-results, a Ridge-regression was run and plotted instead of the raw results.

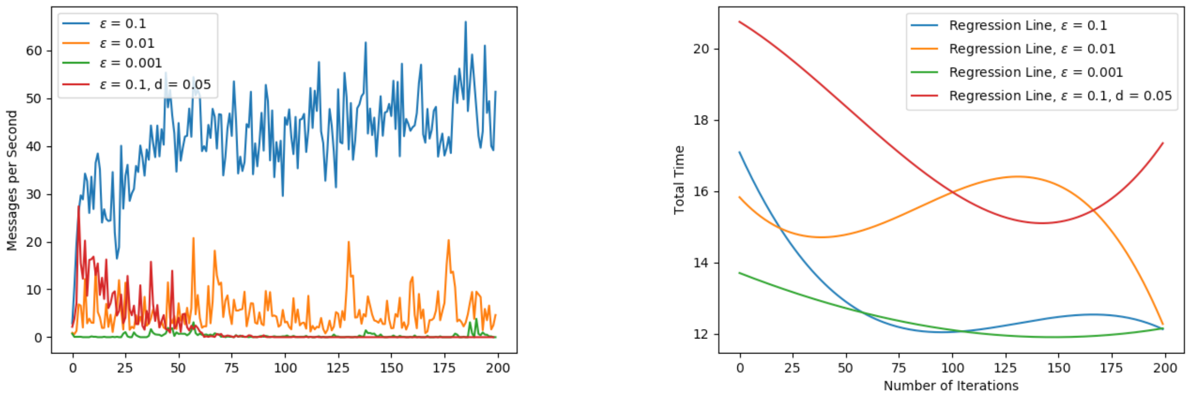


Figure 8 Performance of learning with different exploration rates for an r_{max} of 1200 (time), 500 fps

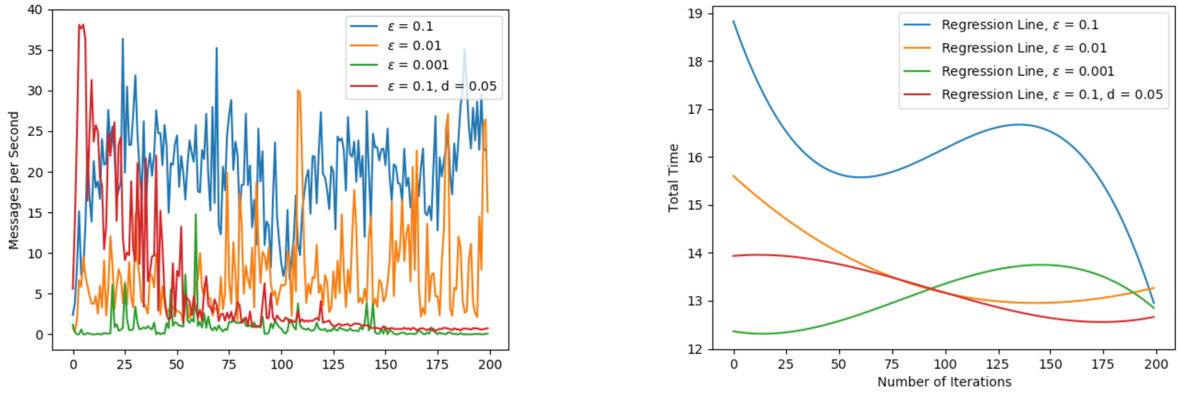


Figure 9 Performance of learning with different exploration rates with r_{max} = 500 (time), 500 fps

From these figures, it can be seen that using lower exploration rates indeed ensures that fewer messages are sent, while generally maintaining a high performance (small time to finish sequence). This shows that agents indeed learn to communicate only the messages necessary. Overall, it seems like the performance is slightly better when r_{max} is higher, although the amount of messages sent is slightly higher. Lower exploration rates as well as a decaying exploration rate generally also seem to cause more stable performance rates.

However, agents in this case optimize their performance only for the specific ‘Rainbow’ map. In the real world, the task might be more diverse. Therefore, simulations were re-run with a ‘Random’ map, in which the task as well as the distribution of blocks randomized every new game. Also, in the previous runs, learning was done with a reward of time. The simulation was sped up, but if we want to generalize the learning mechanism as well as results to a context with humans, in which the speed will be much lower, it is necessary to learn with simulation steps, which remain the same even though the speed changes. The reward had to be balanced out again too, and now values of 1200 and 2500 were used for r_{max} while using simulation steps instead of time to calculate r.

Also, to be able to reliably evaluate the performance of the different sets of parameters, a baseline was established with a non-adaptive communication strategy of communicating nothing (silence). The performance of this baseline strategy can be seen in Table 1. The learning algorithm had to perform better than this baseline, while using a small amount of messages and qualitatively learning explainable behavior.

Table 1. Performance of the baseline run silence

Communication Strategy	**Always Silent**
Mean Time (s)	61.64
Standard Deviation	7.62
Lowest Time (s)	43.40
Highest Time (s)	89.0

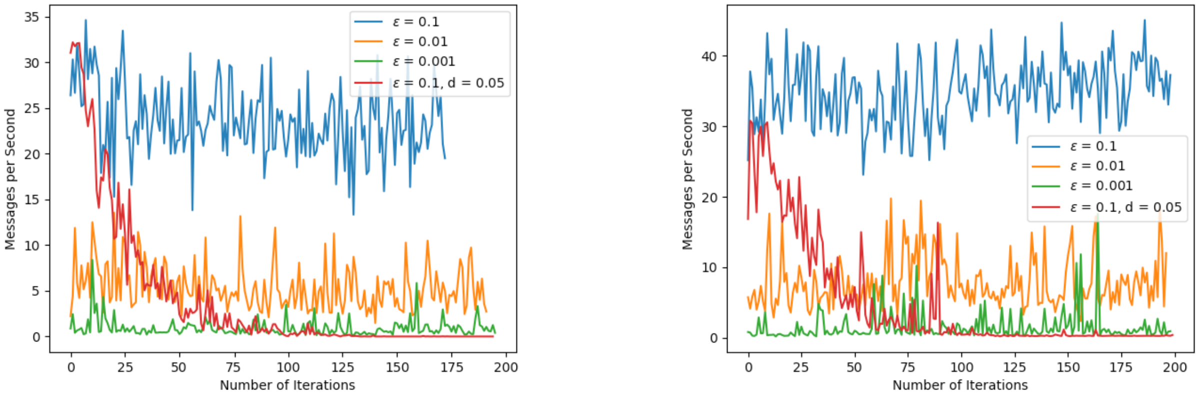


Figure 10. Amount of messages the agents used with r_{max} = 1200 (left) and r_{max} = 2500 (right) in a Random map

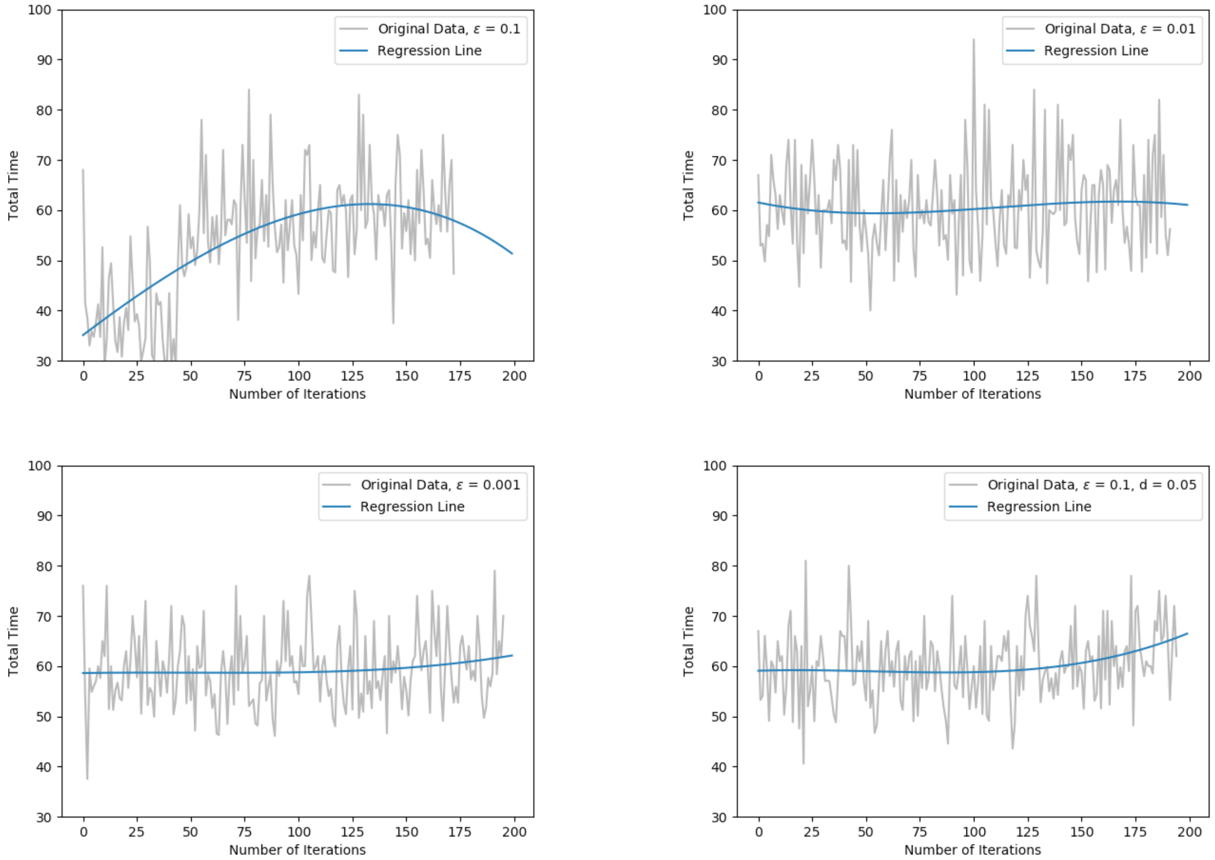


Figure 11. Performance of the agents with r$_{max}$ = 1200

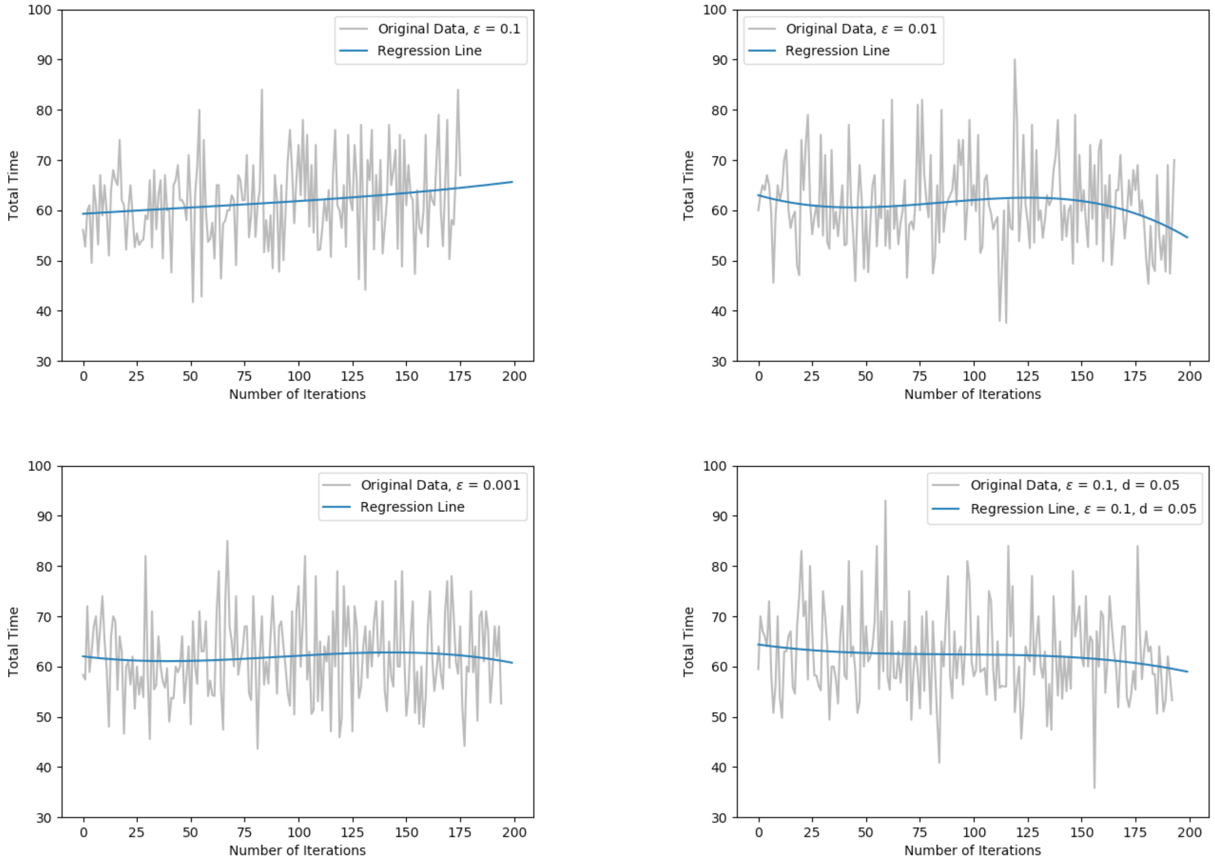


Figure 12. Performance of the agents with r$_{max}$ = 2500

Table 2 Mean performance of the different communicaiton strategies for belief communication. Values with a star were calculated with only the values for the second half of the runs

Height of Reward	Exploration variables	Mean time	Std time	Nr of messages	Std nr of messages
r_{max} = 1200	ε = 0.1	53.63	12.26	1268.22	367.73
	ε = 0.01	60.53	9.17	329.57	168.72
	ε = 0.001	59.25	7.2	54.71	58.43
	ε = 0.1, d = 0.05	59.98 (60.71*)	7.12 (11.53*)	244.42 (5.43*)	457.70 (11.98*)
r_{max} = 2500	ε = 0.1	61.62	8.33	2467.17	363.28
	ε = 0.01	61.12	8.79	532.73	248.75
	ε = 0.001	61.92	8.45	104.07	145.55
	ε = 0.1, d = 0.05	62.27 (61.79*)	8.66 (8.47*)	326.72 (22.6*)	551.12 (12.72*)
	Always Silent	61.64	7.62	0	0

From Table 2, it can be seen that that the learning agents with the low r_{max} perform better than the silence baseline on average, meaning that they are able to use communication to perform better at the task. The agents with the higher r_{max}, however, do not. Looking at the plots, though, it seems as though the learning curve of the agents with a high r_{max}, is more promising, as it has more of a decline. This means that more training rounds would be necessary to achieve a good result. The amount of messages seems more reasonable for the lower r_{max} as well, especially giving a low amount in the second half of the runs in the condition that uses decay of the exploration rate. The average performance however does go up somewhat compared to the average of the full runs, meaning that the agent might actually learn to communicate too few messages. As a general remark, the learning curves seem much flatter than the learning curves in the first few plots. This is due to two aspects: training with a the 'Random' map makes the task performance vary more, and since the overall time is higher due to training at a lower speed, differences appear to be relatively flatter. Overall, this makes it hard to read the results from the plots. The table as well as the qualitative results below give clearer insights, especially when combined.

4.3.2 Qualitative Analysis

Apart from looking at performance and number of messages it is relevant to qualitatively look at the Q-values that the agent has learned as well as the communication behavior. Especially in preparation for an experiment with human participants it is valuable to check whether the used agent behaves in a somewhat reasonable manner.

Table 3 Q-table, sections with the highest expected reward, r_{max} = 1200

Communicated	**In Task**	ε = 0.1	ε = 0.01	ε = 0.001	ε = 0.1, d = 0.05
False	**False**	Communicate	Silence	Communicate	Silence
	0	Communicate	Silence	Communicate	Silence
	1	Silence	Communicate	Communicate	Silence
	2	Silence	Communicate	Communicate	Silence
	3	Silence	Silence	Communicate	Silence
	4	Communicate	Communicate	Communicate	Silence
	5	Communicate	Silence	Communicate	Silence
True	**False**	Silence	Communicate	Silence	Silence
	0	Silence	Silence	Silence	Silence
	1	Silence	Silence	Silence	Silence
	2	Silence	Silence	Silence	Silence
	3	Silence	Silence	Silence	Silence
	4	Silence	Silence	Silence	Silence
	5	Silence	Silence	Silence	Silence

As can be seen from the table, it is generally learned that information should not be shared more than once. Interestingly, the learning algorithm is not able to reliably learn that it should not communicate a block that is not in the sequence. Also, the different models appear to disagree on the best moment to communicate a block, as they all decide 'communicate' as an action for different values of the inTask() state factor. This is to be expected, as the agents have perfect memory and it should therefore not matter when exactly a block is communicated, as long as it is at some point. The condition that used decay of the exploration rate unfortunately learns to communicate nothing at all. This showed in the task performance results, as the average performance went up for the second half of the run, when the exploration rate starts approaching 0. Optimal performance with a small amount of messages is therefore likely to appear with an exploration rate lower than 0.001, but higher than 0.

5. Learning to Communicate Goals

Communicating beliefs or general observations about the environment helps to achieve better performance at a team task, as the previous chapter has shown. It improves the shared situational awareness, and might even help agents and humans to understand the actions of another agent or human. However, it does not enable them to coordinate those actions before they are done. In order to coordinate, communicating the tasks (to be) executed gives a great advantage over not communicating them. Communicating such information enables agents to actively and explicitly coordinate their tasks, even without deliberately discussing the topic; once one agent communicates that it is going to perform some subtask, the other agents can assume that they can pick up some other subtask instead of attempting to perform the same subtask.

In this chapter, we move to another level of communication complexity, namely level C5, to be able to look into what it takes to learn how to communicate goals. With this level of communication complexity, it follows that the agent needs to be more complex as well, as it should be able to reason about actions the other agent will be doing and how they relate to the common goal, as is the case in level A2. The process of developing and evaluating the learning of proactively communicating goals will be discussed.

5.1 Adapting to the Communication of Goals

5.1.1 Problem Description

The problem of learning when to communicate goals has many similarities to the problem of learning when to communicate beliefs. All complications mentioned in the previous chapter still hold. This is also the reason that communication complexity level C4 is not explored; in order to deal with the delayed rewards, a continuous communication decision is necessary. There are however some differences that need to be dealt with.

- **Strong effect of time**: while beliefs about blocks are pieces of information that simply exist (although the information might change when the environment changes), information about goals exists for a while to always be removed at some point. When an agent achieves a certain goal state, that goal is deleted. The same goal can reappear, but the different instances of the goal must be distinguished as the context in which they are adopted might be different.

- **Sequential task-related information**: while information about blocks exists in its own right and is not connected in any way to information about other blocks, different goals are highly connected with each other. Sub goals that lead towards one large goal exist sequentially; they follow each other in a certain order, where one goal might imply another one has been completed or that another one will follow. It becomes even more complex when parallel goals exist within this sequence. This has consequences for the communication of those goals, as (especially human) team members might make inferences about past or future goals from current ones.

Both of these factors have been looked at while exploring the problem of proactively communicating goals, in extension of the existing 'belief-learning' system. In the next section, a more extensive explanation of their implementation will be discussed.

5.1.2 Attempting to Learn Goal Communication

As mentioned before, for the learning of communicating goals, communication complexity C5 and agent complexity A2 was used. This means that a communication decision is made at every cycle *c* for every goal γ that exists in the goal base.

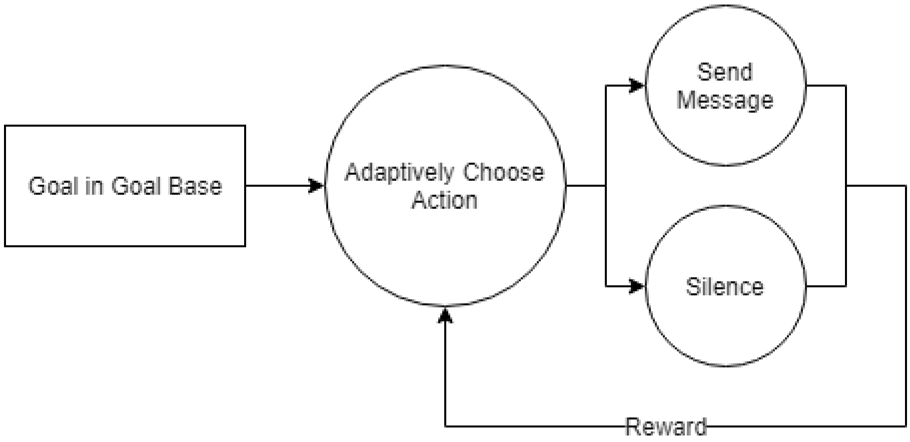


Figure 13. C5A2 Deciding about communication when having a goal

The same learning mechanism and reward as in the belief-communicating case was used. However, in order to define which message should be sent for which goal, a small extension had to be made that checks for the type of goal about which a communication decision is to be made. A level of abstraction is then assigned to every goal, based on how many 'goal-steps' it is away from actions that directly contribute to the main goal. To give an idea of the different possible abstraction levels, a goal to be in the Dropzone while carrying a block is only one step away from dropping a block, which contributes directly to finishing the sequence (the main goal). A goal to pick up a block that an agent has found, is one step further away. A solution for the *sequential task-related information*-problem is to determine such levels of abstraction in rules, and to then make the levels a factor in the state representation of the learning algorithm. That way, the agent can learn which sub goals are important enough to communicate, and which are not. Together with a state factor that checks if the goal has been communicated before, we have our first state representation for learning goal communication, consisting of the general state factors (a) redundancy and (b) abstraction level. They can be translated to beliefs fitting the BW4T environment and GOAL language as follows:

```
communicatedThis(Boolean)
abstraction(Level)
```

It is important to note that, as explained above, a goal that has been dropped and adopted anew later on in a different context cannot always be considered the same goal. For example, if an agent has a goal to pick up a red block for the second block in the sequence, it might later on adopt a goal to pick up a red block for the fifth block in the sequence. This is then essentially new information that might need to be communicated. Therefore it is not always useful to keep a belief about whether a goal has been communicated after the goal has been achieved or dropped. In the current program, every time a goal has been achieved or dropped, any possible beliefs about whether it has been communicated will be deleted. In the future, it might be useful to remember whether a goal has been communicated in relation to a specific context.

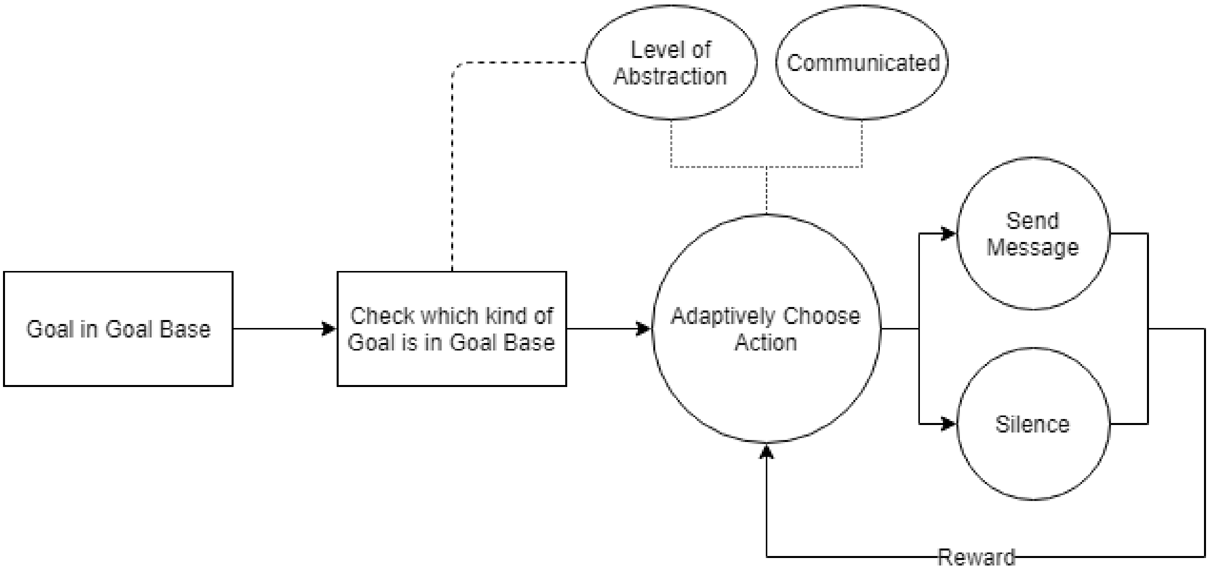


Figure 14 C5A2 Extended (with state factors: dotted lines indicate where state factors are determined and fed into the learning model)

5.2 Leveraging Rules and Learning for Goals Communication

In developing the right state representation for the learning of goal communication, there are several abilities that can be considered for integration in the state representation:

- **Task Coordination**: Understand that a task is performed by someone else and does not *need* to be done by you
- **Capability and Capacity Reasoning**: Reason about who might be best suited for a task and knowing when you are the best agent for performing a certain task
- **Task Reallocation**: Consciously commit to and communicate about a certain goal even though another agent has already done so before to overrule and change the current task division (e.g. when you believe you are better suited to the task)
- **Resolving Conflict**: React to and solve a situation where two agents have expressed commitment to the same goal

These suggestions imply different levels of reasoning complexity that could be implemented in an agent. Also, the more complex abilities drift away from the basic 'goal communication' case, and enable some sort of negotiation about task division, effectively coupling the communication of a goal to a certain kind of commitment that can be negotiated about. In principle, GOAL has a blind commitment strategy implemented, which means that agents will always pursue a goal until the goal state is reached. When communication starts interfering, it changes into blind commitment *if and only if* beneficial to the task performance of the team. If there is a reason to believe it is more beneficial to let another agent perform a task, for instance because of a received message stating that very fact, goals can be dropped.

A question arises about how to implement such abilities. Since we are talking of a hybrid agent that uses both reasoning rules as well as data-driven machine learning, the above rules could sometimes be both explicitly programmed or implicitly taken into account by converting them into variables for the state of the learning system. Underneath, an exploration of the possibilities for implementation of all above mentioned abilities is given.

1. Task Coordination

Since this ability needs active manipulation of goals and beliefs, it can most easily be implemented in rules. An agent could for example drop a goal if someone communicates that it is already working towards that goal, and then automatically move forward to the next step in the task (in the case of BW4T pursuing the next block in line). Transferring it into learning about communication would create a belief such as 'communicatedByTeammate(Boolean)'. While this is a relevant feature to take into account for learning whether you should communicate it too, it does not do anything to the actual goal adoption unless accompanied by a rule. If the rule exists, on the other hand, 'communicatedByTeammate' is not likely to ever become true since then the agent will have already dropped the goal when the other agent communicates it.

2. Capability and Capacity Reasoning:

This feature becomes relevant when communicating something is considered as commitment to a certain task division within the team.

The simplest way to implement this, is to define a reasoning rule that calculates which agent is best suited for the task. This could then be transferred into a belief like 'bestSuitedForTask(Agent)' that could be a feature of the learning process. An advantage of this would be that the programmer has a lot of control over the learned behavior, as well as over the aspects that determine what makes an agent best suited. A problem, however, might be that sometimes the programmer cannot always know what makes an agent best suited in terms of the effect on the task performance.

Exactly this problem might be solved by using more of the power of the learning algorithm. Factors that determine whether an agent is better suited to do a task than another agent could be added to the state representation without abstraction through a knowledge rule. Examples of such factors might be the agent's ability to see colors, their distance to a block, or the current level of its battery. This becomes beneficial when dealing with fuzzy parameters such as 'current workload' of the team members or 'experience with the task'. Such fuzzy parameters will particularly make a difference when playing with a human team member. Also, sometimes uncertainty will play a role, as an agent cannot always know everything about the abilities and current capacity of its team members. In such cases, the learning algorithm's ability to deal with a problem probabilistically will definitely be an advantage.

3. Task Reallocation:

Reallocating tasks is important when the different agents do not always know which agent is the best candidate for performing a certain task. It might happen at some point that an agent believes it is best suited for a task and therefore it committed to performing the task by letting the others know, even though some other agent might know that it is actually easier for him to perform the task. This can happen because team members do not always know where the other agents are located, what they are doing, etcetera. It is closely related to the above feature of capability reasoning, and generally the same mechanisms come into play. The difference, however, is that it must be taken into account how long ago someone has committed to doing a certain task, how far he has progressed, and whether it would therefore be beneficial for another agent to take over.

Deciding whether something is probably beneficial to do is based on very vague premises. Different combinations of situations might bring about different results, and therefore it would work best to implement extra factors in the state representation.

However, it must be accompanied by a rule that makes sure an agent is actually able to overwrite the commitments of another agent. An agent must first decide to ignore the commitment of the other agent, which must be defined in a rule, after which it can decide to communicate its own commitment to let the other agent know that it is not following the proposed task division by the other agent. A way in which this might be implemented is by work agreements and protocols as specified in HATCL, a communication language made by TNO to manage the gap between humans and agents, which can specifically be used for teaming functions (van der Vecht, van Diggelen, Peeters, Barnhoorn, & van der Waa, 2018).

4. Resolving Conflict:

Trying to resolve conflict through communication greatly complicates the communication in general. While it makes coordination of tasks and thus high level goal communication much more valuable, it actually moves out of the current level of communication complexity that we are working with, to level C8, where communications other than simply an agent's own goals or beliefs are possible. In order to be able to resolve conflict, an agent must be able to use explicit communication like asking for questions or giving answers to questions. This could be done through rules, but to proactively use such communication it should be learned as well.

A lot of work has been done on automated negotiation (Baarslag et al., 2013; M. Lewis, Yarats, Dauphin, Parikh, & Batra, 2017; Traum, Marsella, Gratch, Lee, & Hartholt, 2008), therefore it will not be dealt with further in this thesis.

While most of the above factors have not been implemented, an attempt at realizing capability reasoning has been made by adding a factor to the state representation. Since in the current scenario both agents in the team are supposedly the same in terms of their own capabilities, the only difference in how well a certain agent is suited for a task can be expressed by how far they are away from a block they are pursuing. While they cannot decide based on this whether they will pursue it or not, they can decide to communicate their commitment to the specific block to make sure the other agent will not commit to it. The resulting state factor was the following:

```
pursuingBlock(Int/Boolean)
```

The variable in the state factor would be false if the agent would just be exploring the rooms instead of pursuing a specific block, whereas it would be an integer representing the distance between the agent and the pursued block if it was actually pursuing a specific block. The knowledge rule to determine this was as follows:

```
    pursuingBlock(Distance):- roomhasWantedBlock(Room), at(Place), closestRoom(-
Place, Room), setof(Path, path(Place, Path), Set), predsort(sortHelper, Set,
[ShortPath|_]), length(ShortPath, Distance).
```

This knowledge rule basically means that if an agent knows rooms that have the current wanted block, it should take the closest of those rooms to its current location and determine the shortest path to that room. The length of that path is the value of the variable in the 'pursuingBlock' factor.

5.3 The Influence of Exploration and Rewards

As in the belief-communicating case, the agent sends out a large amount of messages, and it is necessary to make sure that the agents learn to communicate just enough to achieve the highest performance. Once more, a baseline was established with a communication strategy of communicating nothing at all, of which the result can be seen in Table 4.

Table 4. Performance of the baseline run silence

Communication Strategy	**Always Silent**
Mean Time (s)	64.44
Standard Deviation	9.16
Lowest Time (s)	44.83
Highest Time (s)	88.0

5.3.1 Quantitative Analysis

Again, the values used for r_{max} were 1200 and 2500. Ridge regression was used to clearly see the development of the performance over time. Also, when communicating goals, there is a chance that the agents will make a fatal mistake, causing them to not be able to finish the sequence. For example, if one of them communicates that they will pick up a block in a certain color, the other agent will skip that block and move to the next. If it happens that that block is closer by, it might happen that the agent accidentally throws away a block, sometimes making it impossible to solve the task. For that reason, the data in the plots below does not reach till 200 rounds most of the time.

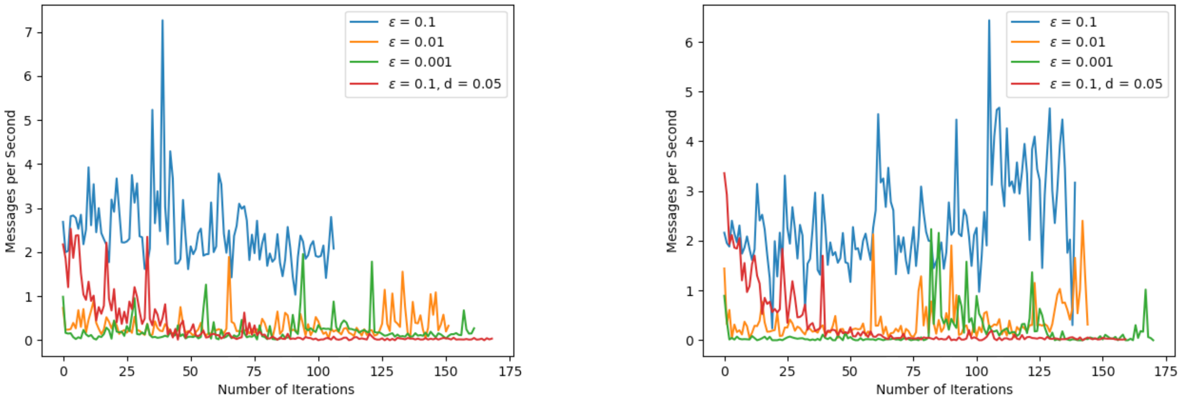


Figure 15. Amount of messages the agents used with r_{max} = 1200 (left) and r_{max} = 2500 (right)

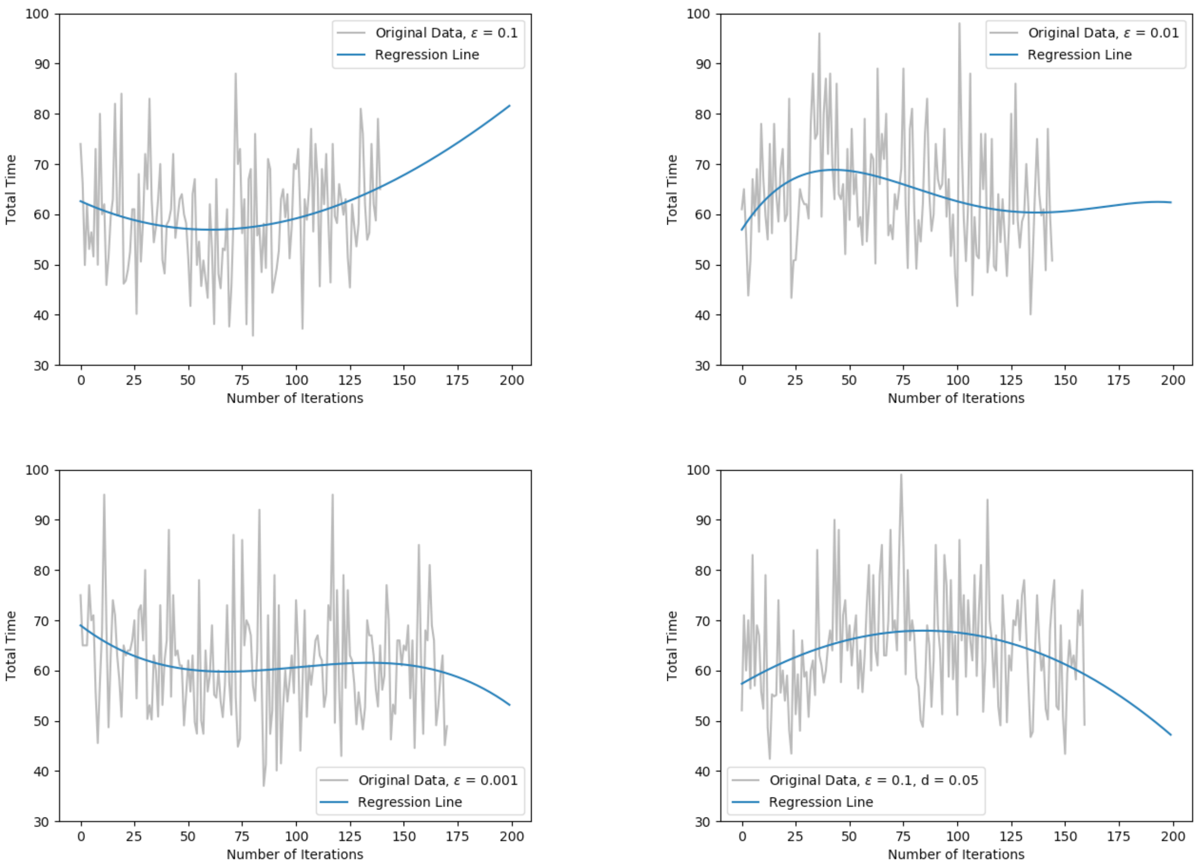


Figure 16. Performance of the agents with r_{max} *= 1200*

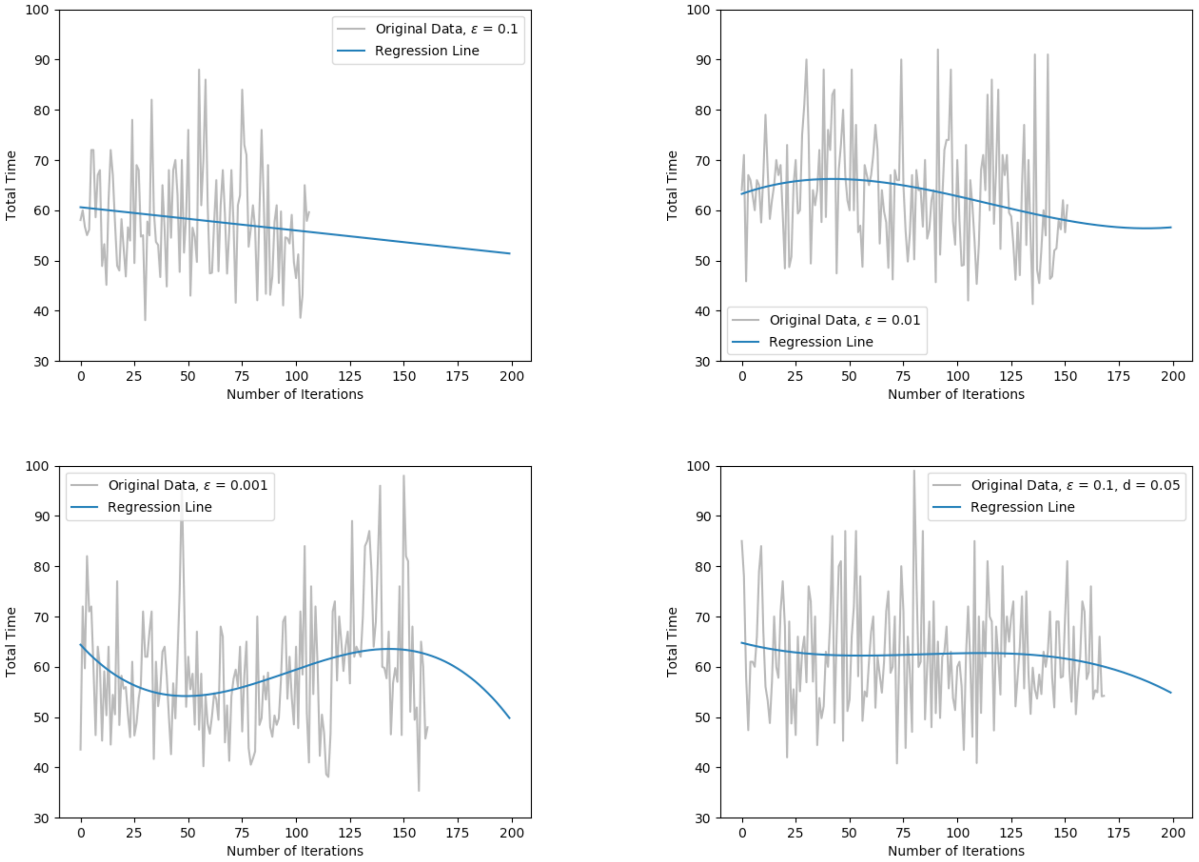


Figure 17. Performance of the agents with r_{max} *= 2500*

Table 5. Mean performance of the different communication strategies for goal communication. Values with a star were calculated with only the values for the second half of the runs

Height of Reward	**Exploration variables**	**Mean time**	**Std time**	**Nr of messages**	**Std nr of messages**
r_{max} = 1200	ε = 0.1	59.40	10.59	140.45	61.98
	ε = 0.01	64.30	11.70	23.57	25.80
	ε = 0.001	61.47	10.89	9.84	19.54
	ε = 0.1, d = 0.05	64.66 (64.18*)	10.93 (10.55*)	19.24 (2.5*)	35.76 (2.6*)
r_{max} = 2500	ε = 0.1	58.15	10.44	142.52	51.29
	ε = 0.01	63.47	11.42	20.34	16.40
	ε = 0.001	58.92	12.35	12.09	16.25
	ε = 0.1, d = 0.05	62.46 (61.74*)	10.82 (9.48*)	19.42 (2.08*)	33.98 (1.54*)
	Always Silent	64.44	9.16	0	0

From Table 5, it can be seen that almost all learning agents are able to, on average, perform better than a silence baseline. This means that they can indeed use communication about their goals to increase their speed. When looking at the plots and numbers, it is clear that the higher reward generally achieves a better result. However, with a lower reward the results are a bit more stable. Also, a lower exploration rate does not always maintain the good results. It can be seen that an exploration rate of 0.001 achieves the best result while keeping the amount of messages low, making sure there is no communication overload. The average amount of messages, 12.09, seems like a number that makes sense. In a situation where there is a task of 6 blocks and agents communicate which block they will be getting, every agent might deliver about half of the blocks. That results in 3 to 6 messages, considering the goal of getting a certain block can be communicated while the agent is going to the room where the block is, as well as when it is actually picking it up. Adding messages for going to the drop zone makes that around 9 messages. Since there can be messages about rooms the agent goes to for exploration, which probably happens between 3 to 9 times, an amount of about 6 to 18 would be a decent amount of messages.

5.3.2 Qualitative Analysis

A qualitative analysis of the learned model of the agents using an r_{max} of 2500 will be given below. The tables show the states in which the learned model indicates it is best to perform a 'communicate' action. The orange cells indicate states in which the information has already been communicated before, and can therefore be seen as redundant.

Table 6. States in which the learned model decides 'communicate' for an exploration rate of 0.1

ε = 0.1		
abstraction(3)	communicated(true)	pursuingBlock(2)
abstraction(3)	communicated(true)	pursuingBlock(8)
abstraction(4)	communicated(false)	pursuingBlock(7)
abstraction(4)	communicated(true)	pursuingBlock(7)
abstraction(4)	communicated(true)	pursuingBlock(false)
abstraction(5)	communicated(false)	pursuingBlock(3)
abstraction(5)	communicated(false)	pursuingBlock(4)
abstraction(5)	communicated(false)	pursuingBlock(9)
abstraction(5)	communicated(true)	pursuingBlock(7)
abstraction(5)	communicated(true)	pursuingBlock(8)

Table 7. States in which the learned model decides 'communicate' for an exploration rate of 0.01

ε = 0.01		
abstraction(3)	communicated(false)	pursuingBlock(false)
abstraction(4)	communicated(false)	pursuingBlock(7)
abstraction(4)	communicated(false)	pursuingBlock(false)
abstraction(5)	communicated(false)	pursuingBlock(3)
abstraction(5)	communicated(false)	pursuingBlock(4)
abstraction(5)	communicated(false)	pursuingBlock(6)

Table 8. States in which the learned model decides 'communicate' for an exploration rate of 0.001

ε = 0.001		
abstraction(3)	communicated(false)	pursuingBlock(9)
abstraction(3)	communicated(false)	pursuingBlock(false)
abstraction(3)	communicated(true)	pursuingBlock(false)
abstraction(4)	communicated(false)	pursuingBlock(6)
abstraction(4)	communicated(false)	pursuingBlock(9)
abstraction(5)	communicated(false)	pursuingBlock(5)

Table 9. States in which the learned model decides 'communicate' for an exploration rate of 0.1 with a decay of 0.05

ε = 0.1, d = 0.05		
abstraction(3)	communicated(false)	pursuingBlock(4)
abstraction(3)	communicated(true)	pursuingBlock(7)
abstraction(4)	communicated(true)	pursuingBlock(8)
abstraction(5)	communicated(false)	pursuingBlock(6)

Looking at these tables, it is interesting to see that there are quite some redundant states left in which the agent communicates for the highest exploration rate. This can be explained by the fact that in the BW4T scenario, the agents currently do not have many and diverse goals. This means that, due to the high exploration rate, the agent is likely to communicate all its goals anyway. Also, since there is only one goal that the agent pursues at the time, communicating the goal is almost always an advantage.

Across the different sets of hyperparameters, it can be seen that there are several in which goals of abstraction level 4 must be communicated when it has not been done before and when the agent is pursuing a block that is still quite far away (6 – 9 steps). A level 4 goal is a goal to go to a room, while the agent knows that the room contains a block that it needs. An explanation for the learned behavior could be that while it is useful to communicate that an agent will get a block from a certain room, if they are too close by, it is better for them to wait until they are actually in the room and then communicate their level 3 goal (which is a goal to be at a block to pick it up in the same room).

In the learned models with the higher exploration rates, it is also noticeable that there are many states which deal with level 5 goals. Level 5 goals are goals to be in a room, while the agent does not know where the next block in the sequence is. Telling your team members to make sure they will not attempt entering the same room apparently helps to increase performance.

Overall, however, it is quite hard to find regularities and rules in the state representations. This could be because knowing when to communicate goals requires a much more subtle understanding of context than communicating mere information. It is necessary to for example take into account where the other team member might possibly be, what he is doing and what his goals are, how long it will probably take before he reaches a certain location, all those factors for the agent itself as well as the relation between the team members on all those factors. The agents are currently not able to reason about all these factors. The current state representation might therefore be too simple to allow for smooth learning, and there is too much variation in the performance.

6. Moving to Human-Agent Teaming

Since the research questions in this thesis are about human-agent teaming, and not about agent-only teams, it is essential to evaluate the developed learning agents in a human-agent setting. This contributes to answering the main research question, as well as answer SQ3, SQ4 and SQ5.

This chapter therefore proposes an experiment set up to evaluate how well human-agent teams perform, as well as how the human side of the team qualitatively experiences the different agents used.

6.1 From Agent to Human-Agent: Experiment Aim and Setup

The main aim of the experiment was to test how results obtained in agent-agent communication experiments relate to human-agent communication on several aspects, as well as to compare the performance of human-agent teams working with three different kinds of agents. This was tested in a scenario where humans solve the BW4T task together with an agent that has been trained in simulation, to enable learning from the human. The experiment thus enabled data collection and evaluation of the following factors:

- **Quantitative task performance**: it was tested how quickly a human-agent team can finish the task, to see if there were any differences in task performance between teams using different kinds of communication.

- **Qualitative teaming measures**: in agent-only teams, there are no qualitative measures. When humans get involved, however, factors such as trust and usability become important. Measuring such factors can help to identify why human-agent teams do not perform similar to agent-only teams. It also helps to get a general insight into important aspects for the future design of proactively communicating agents, as well as providing inspiration for defining the state representation with which the agent learns.

- **Evolution of the learned agent model**: from training in simulation, it can be seen what kind of behavior and rules the agent learns. However, this behavior might change when playing with humans, because humans act differently from agents, due to for example imperfect memory of a limited communication bandwidth. By recording the evolution of the learned model, it can be evaluated how large these differences are and how important pretraining in simulation as well as training with humans might be in the current scenario.

- **Relevance of evaluation methods**: one of the initial research questions was about how communication in human-agent should actually be evaluated, since there is a divide in the literature. On the one hand, AI and Computer Science researchers that focus on the development of the agent mostly take quantitative performance measures, whereas Psychology and HCI researchers mostly look at more qualitative aspects such as usability and trust. This experiment enables us to combine measures from both sides to see if interesting inferences can be made from the combination, and to qualitatively evaluate how human participants respond to the different measures.

During the experiment, the agent continued the learning process, and the learned model of the human-agent teams was compared qualitatively with the learned model of the agent only teams. Three groups of human-agent teams (teaming with belief-communicating agent, teaming with goal-communicating agent and teaming with a combined version) were compared on task performance, efficiency, usability and trust. The agents used were pretrained for 50 runs in the belief-communicating condition and for 80 runs in the two other conditions. All agents used an exploration rate of 0.1 with a decay of 0.05, maintaining an exploration rate of 0.001 during the actual experiment.

While the comparison of the different groups serves to validate and evaluate the performance of the different learning models, further qualitative results aim to explore how humans deal with an agent as a team member. Hopefully those insights might be used to shape further research and experiments in communication in human-agent teaming.

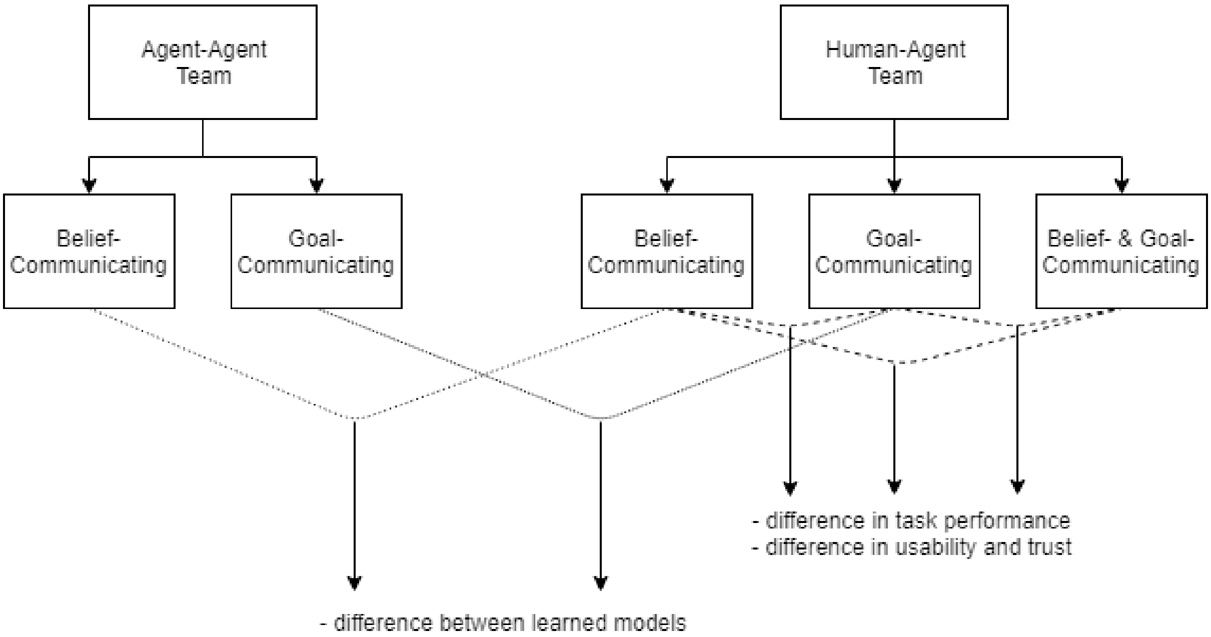

Agent-Agent Team
Human-Agent Team
Belief-Communicating
Goal-Communicating
Belief-Communicating
Goal-Communicating
Belief- & Goal-Communicating
- difference in task performance
- difference in usability and trust
- difference between learned models

Figure 18. Visual of the compared groups

6.2 Experimental Approach and Participants

A mixed-methods approach was used, consisting of interviews, questionnaires and data gathered within the BW4T environment about the performance of the team and the interactions done. The experiment existed of several phases, which will be explained in detail below. The methods used for data gathering will be explained in more detail in those sections as well. Thirty people participated in total (14 male, 16 female), consisting of students from the Technical University of Eindhoven and TNO interns, with an average age of 22 (SD = 2.3). It was assumed that usually people who work in human-agent teams will be trained in working together with agents. The current research is a very early phase, thus it is not necessary to test with a specific user group yet, but using a group that has at least some knowledge about technology and some conception about what an 'agent' is suits the context. Participants were randomly allocated to one of the three groups.

To compensate people for participation, they were given a hot beverage of their choosing as well as some sweet snacks. To motivate them to perform as best as they could, in each condition the person with the best average performance on the task as well as the person with the best single run task performance received a gift voucher of €10,-.

6.3 Hypotheses

Several hypotheses were tested with this experiment.

It was expected that goal communication helps to improve task performance more than belief communication, as has been shown in previous research.

H1 *Human-agent teams making use of goal communication (this includes the combined condition) perform better at the BW4T task than human-agent teams making use of belief communication.*

It was expected that the higher task performance is (at least partly) caused by a higher trust and usability for the goal-communicating agents.

H2 *Human-agent teams making use of goal communication (including the combined condition) perceive higher trust and usability than human-agent teams making use of belief communication.*

The aim of the learning models is to let the agents learn human-specific teaming behavior that cannot be learned in simulation. It is expected that the experiment shows this. It is however not expected that the learned model will have changed completely, since the simulation learning was done keeping the humans in mind.

H3 *The agents will learn different behavior from playing humans when compared to playing in simulation, but they do keep the main learned behavior from the simulation training rounds.*

6.4 Experimental Protocol

Initial Questions

Before starting the actual experiment, some short questions were asked about teaming and teamwork in general, while ensuring that the participant did not know much about the contents of the experiment yet. The questions asked in this short interview were the following:

1. What is your definition of the concept 'team'?
2. What are values that you consider important when collaborating in a team?
3. How should a team member communicate in order to enable a smooth and pleasurable collaborative process?

These questions served mostly to get an idea of a participant's expectations, to make sure they could reflect upon them afterwards.

Experimental Phase 1

The task was explained to the participant, including a walkthrough of the controls and possible actions. No explanation was given about the agent's behavior. The participants had the opportunity to practice with finishing the task three times to get a feel of the game dynamics. After that, the participants performed the task five times in collaboration with the agent. Some questions were then asked about the way the agent's behavior was perceived:

4. Can you explain the agent's behavior?
5. Can you explain how you approached finishing the game? Did you have a particular strategy?

6. On a scale from 1-10, to what extent did you feel you collaborated with the agent as a team?
7. Can you explain why you give this score?
8. What did you think of the agent's communication?

Experimental Phase 2

In the next phase, the mechanisms behind the agent's communication behavior were explained. The participants then had to perform the task another five times in collaboration with the agent. Afterwards, the same questions as in phase 1 were asked to the participant, each time deliberately asking if they gained any new insights.

Evaluating Phase

After these two phases of going through performing the task, a final evaluation phase was seen through. This phase started with the participant filling out a questionnaire about trust and usability of the agent. This questionnaire was based on several existing questionnaires from literature (Bernsen & Dybkjær, 2005; Costa & Anderson, 2011; Jarvenpaa & Leidner, 1999; J. R. Lewis, 1995), but adapted to fit the context of human-agent teaming in the BW4T environment. The questions were evaluated on a Visual Analogue Scale of 10 centimeters. The full questionnaire can be found in Appendix A. Furthermore, some final interview questions were asked:

9. How well did the agent perform as a team member, compared to the values you mentioned in the beginning?
10. What might change in the communication behavior of the agent in order for it to improve?

6.5 Experiment Results

6.5.1 Quantitative Insights

Trust and usability score were both analyzed for differences between the conditions using a One-Way ANOVA. The score for the extent to which people felt they were collaborating as a team and task performance were analyzed using a Repeated Measure ANOVA. If a significant difference was found, a Tukey HSD test was done to test whether there was a statistically significant difference between specific conditions. All values in the tables below which are marked with a star are significant.

Task Performance

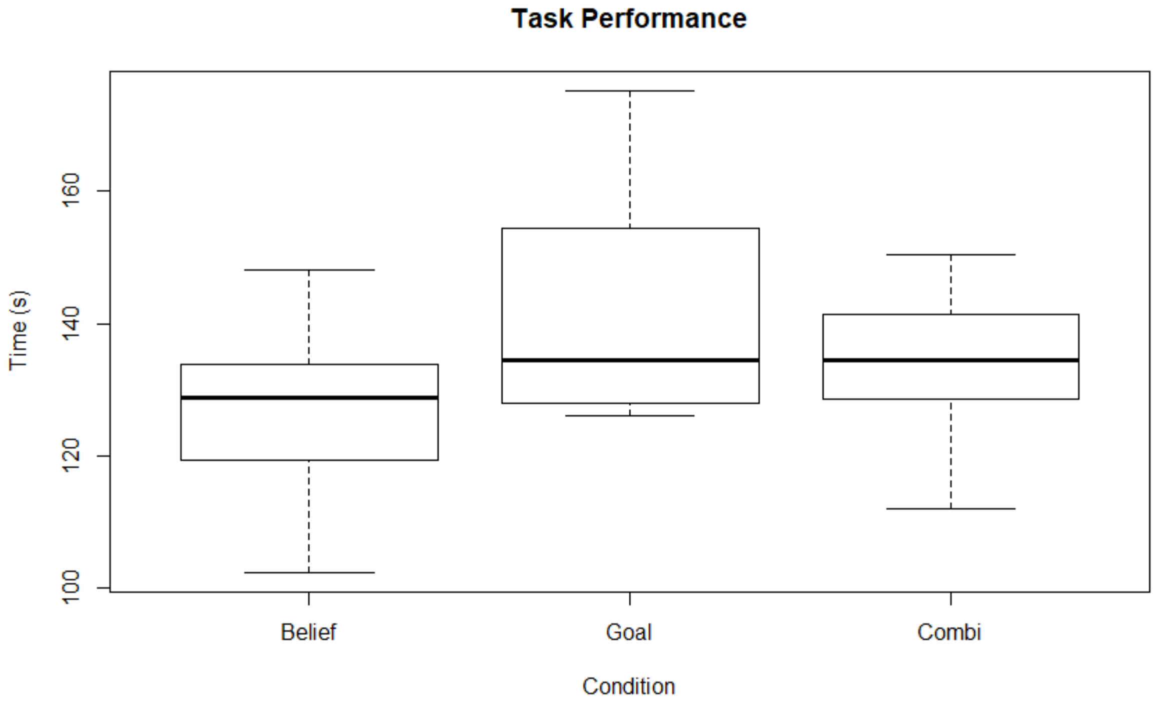


Figure 19. Box plot of the task performance score

Table 10. Results of the repeated measure ANOVA for task performance

	denDf	F value	Pr(>F)
(Intercept)	243	2770.1943	< 0.0001 *
Condition	27	2.4912	0.1016
Game	243	2.0849	0.0311 *
Condition:Game	243	0.4928	0.9598

The ANOVA showed that there is no significant effect in performance between the three conditions of playing with a belief communicating agent, a goal communicating agent or a combined agent, but that there is a significant effect between the different games played. A post-hoc analysis with a Tukey HSD test however finds no significant differences between any of the specific game rounds.

Subjective Team Experience

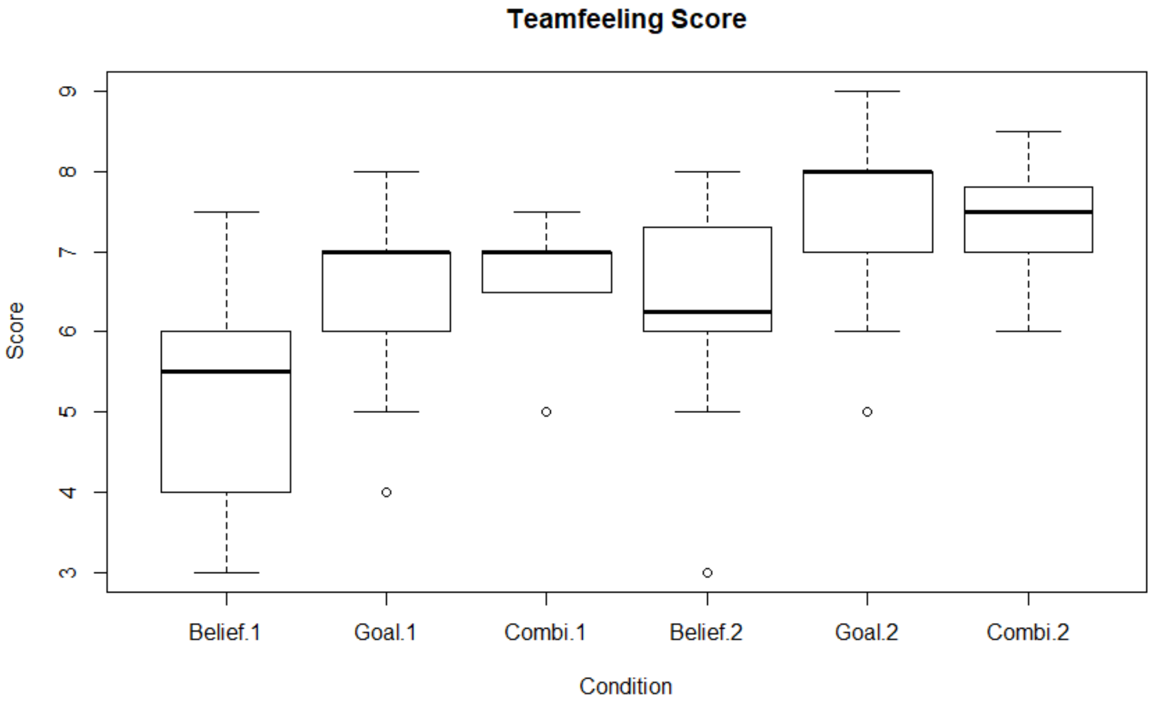


Figure 20. Box plot of the subjective team experience or teamfeeling grade given during the interview, split in phase 1 and phase 2 per condition

Table 11. Results of the repeated measure ANOVA for subjective team experience

	denDf	F value	Pr(>F)
(Intercept)	27	1238.6050	< 0.0001 *
Condition	27	4.7170	0.0175 *
Phase	27	20.4928	0.0001 *
Condition:Phase	27	0.5407	0.5885

Table 12. Results of the Tukey HSD test for subjective team experience

	Estimate Std.	Std. Error	Z value	Pr(>|z|)
Goal – Belief	1.2500	0.5127	2.438	0.0443 *
Combi - Belief	1.4500	0.5127	2.828	0.0140 *
Combi - Goal	0.2000	0.5127	0.390	1.0000

There is a significant effect between the three conditions with p = 0.0179, as well as between the two phases with p = 0.0001. The Tukey HSD test indicates that the mean subjective team experience score for the 'belief-communication' condition (M = 5.78, SD = 1.35) is significantly different from both the 'goal-communication' condition (M = 7.01, SD = 0.98) and the combined condition (M= 7.01, SD = 0.65). The mean for the latter two conditions is exactly the same. The Tukey HSD test shows a significant difference of p = 0.00161 between the scores given in phase 1 and those given in phase 2 as well.

Usability

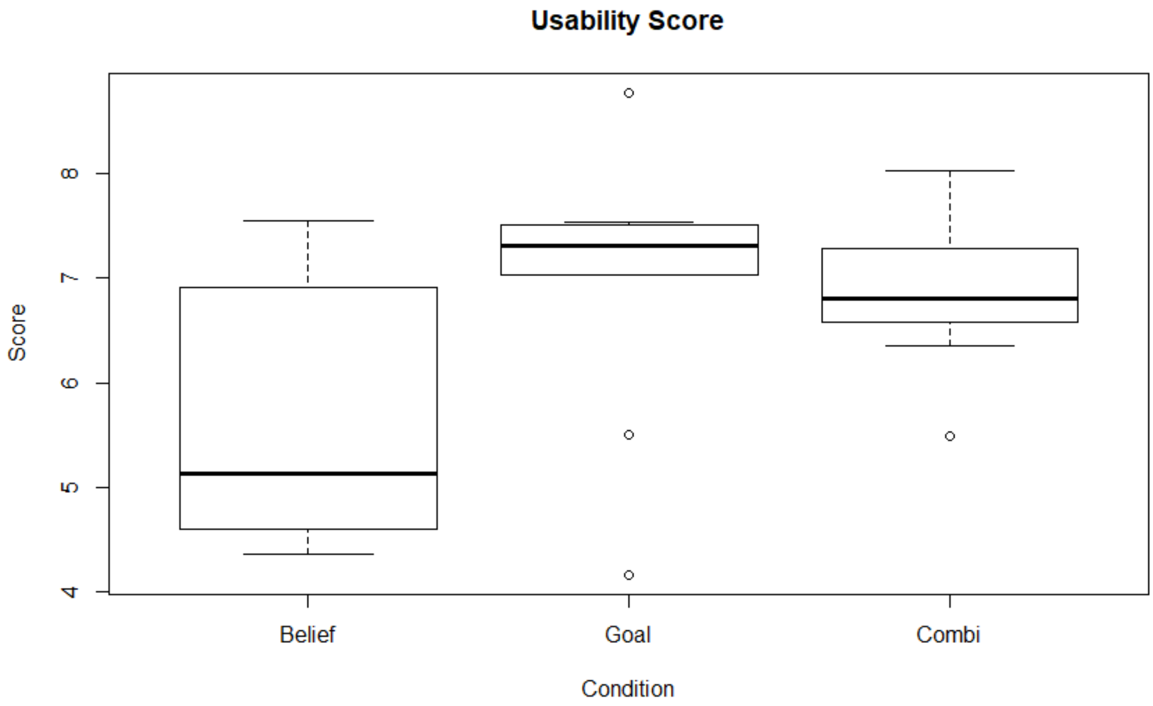


Figure 21. Box plot of the usability score from the questionnaire

Table 13. Results of the one-way ANOVA for usability score

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Between groups	2	12.24	6.12	5.017	0.014 *
Within groups	27	32.94	1.22		

Table 14. Results of the Tukey HSD test for usability score

	diff	lwr	upr	p adj
Goal – Belief	1.3818	0.1571	2.6065	0.0246 *
Combi – Belief	1.3264	0.1018	2.5511	0.3178
Combi - Goal	-0.0553	-1.2800	1.1693	0.9931

The ANOVA showed that there is a significant difference between the three groups for usability score, with p = 0.014. The Tukey HSD test indicates a significant difference between the 'belief-communicating' condition (M = 5.13, SD = 1.23) and the 'goal-communicating' condition (M = 6.97, SD = 1.26). However, the combined condition (M= 6.92, SD = 0.72) does not differ significantly from the other groups.

Trust

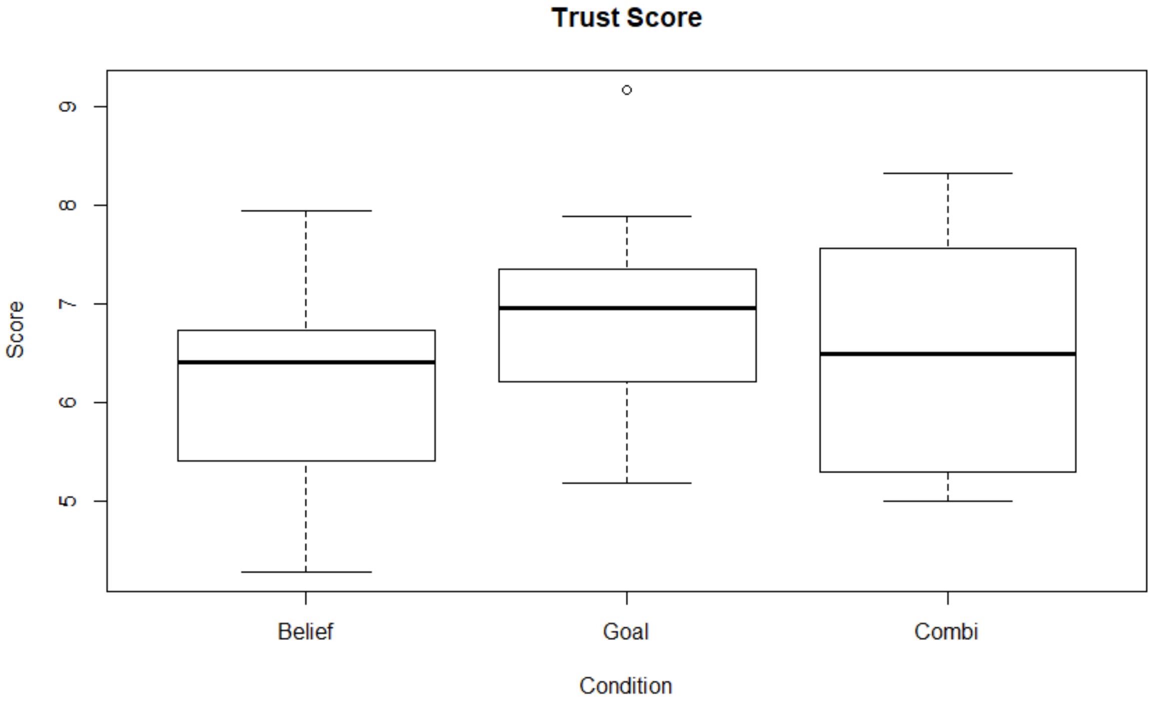


Figure 22. Box plot of the usability score from the questionnaire

Table 15. Results of the one-way ANOVA for trust score

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Between groups	2	3.72	1.862	1.476	0.246
Within groups	27	34.06	1.261		

There is no significant effect for trust between the three conditions, although from the plot it seems like the score of the goal-communication condition is slightly higher than that of the other two conditions. As can be seen from the Sum of Squares value, there is a large variance within the groups. Looking at the raw data, it is clear that the score for the different questions varied greatly, indicating that the questions do not measure trust very reliably for the scenario studied.

6.5.2 Qualitative Insights

Learned Model

The outcomes of the learning model after playing with humans are given below. They are presented in tables with all states in which the agent decides 'communicate' as an action in either the learned model for at least one of the participants, or the base learning model trained only in simulation. Blue cells indicate states that appear in human-trained models, but were already present in the simulation-trained model. Orange cells indicate states that were lost in the transfer from simulation-training to human-training.

Belief-Communication Condition:

Table 16. The different states that decide 'communicate' in the learned model in the belief-communication condition. The appearance row states how often they appear across participants.

State		Appearance
communicated(false)	inTask(2)	5
communicated(false)	inTask(3)	3
communicated(false)	inTask(4)	4
communicated(false)	inTask(5)	3
communicated(false)	inTask(0)	

It can be seen from Table 16 that after playing with humans, the model learns to communicate in an average of 1.5 states. This means that the agent learns that there are 1 or 2 states in which communicating about blocks is most beneficial. It differs per person whether a block that exists in the sequence in position 2, 3, 4 or 5 should be communicated, although communicating when it exists in position 2 appears in half of the participants.

In the transfer to humans, the agents learned to no longer communicate a block that is the current next block in the sequence.

Goal-Communication Condition:

From Table 17 it can be seen that after playing with humans, for each participant agents learn to communicate when: (1) the goal they are pursuing is of abstraction level 3, (2) it has not been communicated before, and (3) they are pursuing a block that is 4 to 8 steps away, or they are not pursuing a block. If agents have a goal of this abstraction level, it means they are already in a room with a block they need. Since there is no existing distance between them and the actual pursued block, if the 'pursuingBlock' variable has a value, it means that there is another block of the same color somewhere in the map. Since this must be in another room, it is probably at least 4 steps away. Therefore, the first five states in the table can be interpreted as an agent basically always communicating when it will pick up a block of a certain color if it is in the room with that block.

Also, the agents learn for each participant that a level 4 goal must be communicated if it has not been communicated before and if they are pursuing a block that is 7 steps away. A level 4 goal is a goal to be in a room, while that room contains the block that the agent is currently searching. The presence of this state can be interpreted as follows: agents should communicate that they are picking up a certain block if they are still quite far away from that block. If they are closer by, it is better to wait until they are in the room with the block to communicate it then.

Last, the agent learns that it should communicate a goal of abstraction level 5 most of the time. A goal of abstraction level 5 is a goal to go to a room when the agent does not currently know where the next block is. The reason that there are still several states in which the agent is supposedly pursuing a block, is probably because a previous block was dropped while the agent was already going to a room, and it does know where the next block in the sequence is. If this happens, and it has not already communicated the goal in the 'pursuingBlock(false)' case, they should still communicate it, to let their human team members know where they are going, especially if the pursued block is close by (e.g. 4 steps away in this case).

Table 17. The different states that decide 'communicate' in the learned model in the goal-communication condition. The appearance row states how often they appear across participants.

State			**Appearance**
abstraction(3)	communicated(false)	pursuingBlock(4)	10
abstraction(3)	communicated(false)	pursuingBlock(5)	10
abstraction(3)	communicated(false)	pursuingBlock(6)	10
abstraction(3)	communicated(false)	pursuingBlock(7)	10
abstraction(3)	communicated(false)	pursuingBlock(8)	10
abstraction(3)	communicated(false)	pursuingBlock(false)	10
abstraction(3)	communicated(true)	pursuingBlock(5)	2
abstraction(4)	communicated(false)	pursuingBlock(5)	4
abstraction(4)	communicated(false)	pursuingBlock(7)	10
abstraction(4)	communicated(true)	pursuingBlock(2)	1
abstraction(4)	communicated(true)	pursuingBlock(9)	1
abstraction(4)	communicated(true)	pursuingBlock(false)	7
abstraction(5)	communicated(false)	pursuingBlock(3)	1
abstraction(5)	communicated(false)	pursuingBlock(4)	10
abstraction(5)	communicated(false)	pursuingBlock(5)	10
abstraction(5)	communicated(false)	pursuingBlock(6)	1
abstraction(5)	communicated(false)	pursuingBlock(7)	8
abstraction(5)	communicated(false)	pursuingBlock(false)	5
abstraction(3)	communicated(true)	pursuingBlock(4)	
abstraction(4)	communicated(false)	pursuingBlock(3)	
abstraction(4)	communicated(false)	pursuingBlock(6)	
abstraction(4)	communicated(false)	pursuingBlock(8)	
abstraction(4)	communicated(true)	pursuingBlock(6)	
abstraction(4)	communicated(true)	pursuingBlock(7)	
abstraction(4)	communicated(true)	pursuingBlock(8)	
abstraction(5)	communicated(false)	pursuingBlock(2)	
abstraction(5)	communicated(true)	pursuingBlock(5)	
abstraction(5)	communicated(true)	pursuingBlock(6)	
abstraction(5)	communicated(true)	pursuingBlock(false)	

Combined-Communication Condition:

Table 18. The different states that decide 'communicate' in the learned model in the combined condition for belief communication. The appearance row states how often they appear across participants.

State		Appearance
communicated(false)	inTask(0)	7
communicated(false)	inTask(1)	3
communicated(false)	inTask(2)	1
communicated(false)	inTask(3)	3
communicated(false)	inTask(4)	3
communicated(false)	inTask(5)	5
communicated(false)	inTask(false)	1
communicated(true)	inTask(false)	2
communicated(true)	inTask(4)	

In the combined-communication condition, the agent learns to communicate information about the location of blocks in an average of 2.5 states. Communicating about a block that is the next one in the sequence (inTask(0)) appears most often. Apart from that, the preference for a communication moment greatly varies between participants.

Table 19 The different states that decide 'communicate' in the learned model in the combined condition for goal communication. The appearance row states how often they appear across participants.

State			Appearance
abstraction(3)	communicated(false)	pursuingBlock(4)	10
abstraction(3)	communicated(false)	pursuingBlock(5)	9
abstraction(3)	communicated(false)	pursuingBlock(6)	9
abstraction(3)	communicated(false)	pursuingBlock(7)	10
abstraction(3)	communicated(false)	pursuingBlock(8)	6
abstraction(3)	communicated(false)	pursuingBlock(false)	10
abstraction(3)	communicated(true)	pursuingBlock(5)	3
abstraction(3)	communicated(true)	pursuingBlock(8)	3
abstraction(4)	communicated(false)	pursuingBlock(5)	4
abstraction(4)	communicated(false)	pursuingBlock(false)	6
abstraction(4)	communicated(true)	pursuingBlock(7)	9
abstraction(5)	communicated(false)	pursuingBlock(4)	10
abstraction(5)	communicated(false)	pursuingBlock(5)	4
abstraction(5)	communicated(false)	pursuingBlock(false)	1
abstraction(3)	communicated(true)	pursuingBlock(6)	
abstraction(3)	communicated(true)	pursuingBlock(7)	
abstraction(4)	communicated(false)	pursuingBlock(3)	
abstraction(4)	communicated(false)	pursuingBlock(4)	
abstraction(4)	communicated(false)	pursuingBlock(6)	
abstraction(4)	communicated(false)	pursuingBlock(7)	
abstraction(4)	communicated(true)	pursuingBlock(8)	
abstraction(5)	communicated(false)	pursuingBlock(6)	

The results of the learning model in the combined-communication condition are very similar to those in the goal-communication condition. Again, the agent learns to communicate a goal of abstraction level 3 almost always, it communicates a goal of level 4 mostly when it is still quite far away from the pursued goal, and it learns to communicate a level 5 goal once a block in the sequence is dropped and the agent knows where the next block is, especially if it is close by.

Interview Insights

Values Important in Teamwork:

In the beginning phase of the experiment, it was asked what people considered important values and how a team member should communicate. As a value, Communication was mentioned by 15 participants. Other important values were Living up to expectations (n = 5), Listening (n = 4), Understanding (n = 4), Trust (n = 4), Equality (n =4), Respect (n = 4) and knowing and using each other's capabilities.

For communication, participants thought that team members should communicate about their actions and plans (n = 14) as well as about possible problems or complications appearing (n = 9). Furthermore, they considered it important that the communication is clear (n = 8), transparent (n = 4) and timely (n = 4).

Some remarkable, though not frequently appearing values or aspects people considered important were the possibility for social bonding (n = 2), having a supportive attitude in general (n = 2) and the possibility for discussion and expressing opinions (n= 2).

Insight in Agent:

Most people understood the basic functionality of the agent already after round one. In the 'belief' condition, people mentioned that the agent communicates everything it sees in a room (n = 5), that the agent explores the rooms one by one in a certain order (n = 5), that the agent starts at the bottom left room (n = 4) and that the agent searches for the first block in the sequence (n = 3). Interestingly, people thought that the agent was much faster than them (n = 4). In the second round, not many participant noticed differences. The most mentioned behaviors after the second round were that the agent communicated irrelevant information (n = 2) and incorrect information (n = 2).

In the 'goal' condition, the participants observed that the agent tells where he is going (n = 6) and that the agent skips a block if the human tells they will get that block (n = 5). However, several also mentioned that the agent does not always communicate everything (n = 4). Just like in the 'belief' condition, some people thought that the agent was much faster than them (n = 2). After the second round, again not many new behaviors were mentioned. Interestingly, someone thought that the agent's behavior was less random (n = 1) and another person thought that the agent communicate more than in the first round (n = 1).

In the 'combined' condition, participants noticed that the agent skips a block if the human tells they will get that block (n = 7). Also, they observed that the agent communicated irrelevant information (n = 6) and that it generally communicates all blocks it finds in a room (n = 5). Some people however also specifically mentioned that the agent communicates relevant information (n = 3). Related to that, one person observed after the first round that the agent tells them the location of a block if they say they will get that block (n = 1). Another person noticed this after the second round (n = 1). Related to this, it was mentioned that after the second round the agent communicated more truthfully (n = 1) and more relevant information (n = 1). Apart from that, participants saw that the agent sometimes drops unnecessary blocks (n = 2).

Strategies:

In the 'belief' condition, participants were quite constant in their strategies. Most people searched for the second (n = 6) or third (n = 3) block in the sequence. They mostly started searching in different rooms than the agent, for example in the top rooms (n = 2), the bottom right (n = 2) or the middle rooms (n = 1). Communicating was only mentioned as a strategy once after the first round, while after the second round several people said that they tried to communicate relevant blocks (n = 5), blocks further in the sequence (n = 1) or just as much as possible (n = 1). Some deliberately mentioned they tried to communicate more (n = 2). Also, one person tried to mislead the agent by communicating there were certain colors of blocks in rooms that were not there, to make the agent move to the second block in the sequence (n = 1).

In the 'goal' condition, the strategy was much less present and uniform. Participants still went for the second block (n = 4), but apart from that there was no real common strategy. After round two, people noticed that they should not communicate that they would get a block if there were more of the same color in a row (n = 2).

In the 'combined' condition, participants again had somewhat more of a strategy. Some people skipped blocks that the agent told them it would get (n = 3) and started in rooms at the right (n = 3) or generally went the opposite way of the agent (n = 2). Participants paid attention to agent communication about goals (n = 2) and blocks (n = 1) and took turn in delivering blocks (n = 2). After the second round, they also decided to communicate more (n = 3).

Evaluation of the Agent:

Participants were both critical and positive about the agent on different aspects. Across all conditions, participants were annoyed by the fact that messages disappeared quickly (belief: n = 6, goal: n = 4, combined: n = 3). In the 'belief' condition, it was also mentioned that the roles were unequal (n = 3) and that the communication is confusing (n = 3), the latter mostly because they were unsure how to deal with messages that were irrelevant to the task. The same amount of people thought it was possible to predict what the agent was doing as people who thought they did not know what the agent was doing (n = 3). Some people mentioned that the communication was useful (n = 2) or fine (n = 2).

In the 'goal' condition, people mentioned that they trusted the agent (n = 4), that the communication was useful (n =

3), clear (n = 2) and that the agent was transparent (n = 2). However, also here, participants did not know where the agent was (n = 2), sometimes missed a personal part (n = 2) and thought there was not enough communication (n = 2).

In the 'combined' condition, participants noticed that there were too many messages (n = 4) and that the communication is confusing (n = 2). However, they also thought it was useful (n = 3), they trusted in the agent (n = 2) and knew what the agent would do (n = 2).

Improvements:

Participants mentioned many possible improvements for both the agent communication. In both the 'belief' and the 'combined' condition, participants indicated that it would be better if the agent communicates relevant information only (belief: n = 6, combined: n = 4). In the 'belief' condition, participants mentioned that they thought the agent should be able to communicate plans (n = 5) and in the 'goal' condition they mentioned that they wanted to be able to communicate information about blocks (n = 3), indicating that both goal- and belief-communication is important.

In general, several extra options for communications were mentioned, where some people indicated they wanted more communication options in general (belief: n = 1, goal: n = 1). Suggestions were the ability to discuss strategy or task division (belief: n = 1, goal: n = 4, combined: n = 1), the possibility to ask questions (belief: n = 1), the possibility to do suggestions for actions (belief: n = 1) and the possibility of telling an agent what not to do (goal: n = 1).

Interestingly, some participants emphasized that they missed a social aspect, in the form of having a kind of avatar or image of the agent (goal: n = 1), more human-like communication (goal: n = 1) or more social communication, like the possibility of giving a compliment (n = 1).

A full overview of all mentioned words and phrases from the interviews across all categories can be found in Appendix B.

6.6 Discussion of Experiment Results

6.6.1 Meaning of Quantitative Results

From the experiment, it was found that the two conditions in which information about goals is shared perform significantly higher than the condition in which only belief-communication is used on both subjective team experience score and usability. No significant effects were found on task performance and trust.

The results on task performance do not match hypothesis **H1** or results in previous studies which indicate that communication about goals increases task performance (Butchibabu, 2016; Harbers et al., 2012; Li et al., 2016)humans must team with agents to achieve joint aims. When working collectively in a team of human and artificial agents, communication is important to establish a shared situation of the task at hand. With no human in the loop and little cost for communication, information about the task can be easily exchanged. However, when communication becomes expensive, or when there are humans in the loop, the strategy for sharing information must be carefully designed: too little information leads to lack of shared situation awareness, while too much overloads the human team members, decreasing performance overall. This paper investigates the effects of sharing beliefs and goals in agent teams and in human-agent teams. We performed a set of experiments using the BlocksWorlds for Teams (BW4T. This is most likely due to the fact that the agents used in the goal-communication and combined condition were more complex than agents previously used. From this, it can be concluded that using goal communication is not always better for the task performance than belief communication. It depends a lot on the capabilities of the agent in dealing with the goal communication, and the interpretation of the humans of both the agent's communication as well as their behavior.

While it is not clearly described, it is likely that previous studies used an agent that could not anticipate on humans communicating their goals, to make sure the behavior of the agents would be similar across conditions (such complex behavior is inherently not present in a belief-communicating condition). By complicating the behavior of the agents, participants were more prone to make mistakes, raising the mean task performance in the conditions where goal communication was used.

Interestingly, however, two of the subjective measures (subjective team experience and usability) still show a significantly higher score in the conditions where goal communication was used. This means that humans qualitatively value agents that communicate their goals higher than agents that just communicate about environment information. This effect is apparently not dependent on objective task performance.

No significant effect can be found for the trust part of the questionnaire. There are two possible reasons for that. First of all, from the interviews it can be concluded that participants generally highly trusted the agent. They did not expect it to lie and they assumed it would do its task, probably due to their comparison with a computer. This means that trust would be high in general. On the other hand, there were large differences between the scores for the different questions. This means that the questionnaire might not be very suitable to measure trust in a human-agent teaming context, as the questions were adapted from questions about a human teaming context. Different factors might be relevant in such a context, meaning that it would be useful to thoroughly evaluate a questionnaire for trust in human-agent teaming in the future.

6.6.2 Insights from Qualitative Measures

Looking at the learned models of the different agents, it is clear that the main behavior that all agents learn, which is to mostly not communicate information more than once, is transferred from the simulation runs to the human runs. Since this is quite basic behavior that we humans would certainly consider to be common sense, it is useful to learn this in simulation rather than in training with humans. In more complicated settings than the BW4T environment, agents might learn that it is useful to sometimes repeat a message if a human has forgotten the information. A state factor that represents how long ago a message has been communicated will be necessary for that.

Some other learned factors transfer from the simulation runs to the human runs as well, but it is hard to explain the exact behavior behind that, as the learned model after the simulation runs does not seem very consistent. However, after playing with humans, it becomes much easier to explain the learned behavior. Possible 'weird' state factors that were present after the simulation runs disappear, while other state factors that are similar to the existing state factors are added (e.g. a state with pursuingBlock(5) existed, and one with pursuingBlock(6) is added). In merely ten games, the agent is able to sharpen the learned behavior and taylor it to a situation in which it is playing with humans.

What is interesting to see as well is that the learned model for communicating beliefs is slightly different in the belief-condition than in the combined-condition. The main difference is the presence of communicating when the state contains 'inTask(0)' in the combined-condition, which means communicating the next block in the sequence. An explanation for this could be that when the agent is able to also communicate that it will get that block, providing the information about the location of the block confirms that the agent knows where the block is and therefore provides extra explanation for its actions. If the agent is unable to communicate that, as is the case in the belief-condition, communicating the location of the block will most likely cause the human to chase after it as well, causing the team to waste time.

The improvement in learned model is however not visible in the task performance, and mostly does not show in the interviews. Only very few people in the combined-condition mention that the agent communicated information exactly when they needed it. In the belief-condition, most people considered the communication of the agent irrelevant at some point, and in the goal-condition, most people were generally positive about the communication. One of the reasons for this quite constant but not very interesting qualitative result, might be the relatively high exploration rate. In the experiments, the exploration rate was 0.001, which seems low, but considering that the agent made the decision to communicate about 20 000 to 30 000 times every game, this still means 200 to 300 random communication decisions. This thus means approximately 100 to 150 extra messages in a game that lasts about 100 to 150 second. This insight emphasizes a general problem in communication learning, which is that exploration is necessary for learning proper behavior, but that exploration also quickly causes problems in a scenario where humans are involved. In future attempts to learn communication, it is therefore advisable to use the simulation runs for exploration runs, and drastically lower it for human runs.

7. Discussion

The aim of this study has been to explore the topic of proactive communication in human-agent teaming and to develop agents that are able to learn when to share which information with their human team. For this purpose, a method that leverages rules and learning to use context information in the decision process has been explored and implemented in agents that communicate about their beliefs and goals. These agents have been evaluated in an experiment in which they played together with humans, where it was found that humans qualitatively evaluate agents that communicate their goals higher than agents that only communicate their beliefs. Also, basic learned behavior from simulation runs transferred to human trials, where agents were able to learn better explainable behavior after playing with humans. In this chapter, the relevance of these results in relation to previous and future research is discussed.

7.1 Relevance and Implications

As has been established in the beginning of this thesis, a choice was made for a hybrid agent to enable learning of proactive communication, because a balance between adaptability and control is necessary in the safety-critical domains in which human-agent teaming is used. The developed agents were BDI-agents using a reinforcement learning module to learn about proactive communication. While the integration of RL in BDI-agents has been used in previous studies (Singh & Hindriks, 2013; Singh, Sardina, Padgham, & Airiau, 2010; Singh et al., 2011), using it specifically for communication has not been done before. Combined with the approach that was taken here, to carefully evaluate the influence of both rules and learning on the final agent behavior, it poses a new way of looking at the development of autonomous agents with learning abilities.

By using the BDI paradigm as a basis for the agents, it is possible to use it as a binding factor between different implementations of agent functionalities such as the learning algorithm used in the current study. The state factors for the learning algorithm were now fully defined by knowledge rules, but they could essentially be full models as well, of which the outcomes are translated to fit the learning mechanism by rules of the BDI-program. To clarify this idea a visual representation of an extensive implementation for learning communication can be seen in Figure 23.

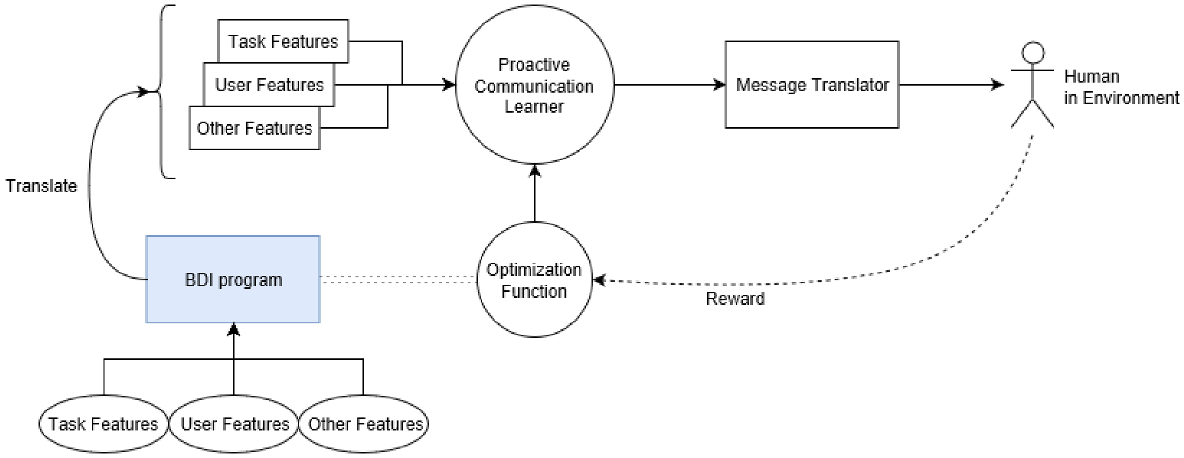


Figure 23. Hybrid implementation of learning communication

In this example, there might be several context features that determine whether information should be communicated, such as information about the user, task context, environment, etc. These features might exist of full models that give complex outputs. These outputs can be fed into the BDI program, which can translate the features to the right level of abstraction to serve as state factors in the learning model. The BDI program therefore manages all model outputs and inputs to make sure that they play together in creating the desired behavior.

A great advantage of this approach is the fact that it allows for modularity. Different models of varying complexity can be added, using the BDI program to make them interact with each other. Next to that, there is a lot of control over the actual learned behavior, as the state representations used for learning can be carefully crafted by the programmer, while the agent can still reason with the factors as beliefs and possibly even goals. This means that the state representation can be built up of a combination of fuzzy data and clear factors and everything in between, allowing the programmer to add in domain knowledge as well. It follows that it is easier to move from a context with only agents playing in simulation to a human-agent context, as the high level of control makes it easier to ensure reasonable behavior, and it helps to use smaller amounts of learning runs to enable online learning with humans. It is in line with the ideas for the future of BDI-agents of (Logan, 2015, 2017)e.g., sensing, deliberation, problem-solving and action, in a single system. There has been considerable progress in both the theory and practice of agent programming since Georgeff & Rao's seminal work on the Belief-Desire-Intention paradigm. However, despite increasing interest in the development of autonomous systems, applications of agent programming are currently confined to a small num-ber of niche areas, and adoption of agent programming languages (APLs as well. In these papers, it is stated that using BDI as a higher level problem description is the best way to use the paradigm in the future. Also, parallels can be seen with the Social Practices approach that was named in the beginning and used as an inspiration for the current work (V. Dignum & Dignum, 2015). In both approaches, context factors are used to limit action possibilities by rules while allowing for a more extensive deliberation process, in this case through Reinforcement Learning, for the final decision.

All of the abovementioned advantages will hold mostly for problems with similar characteristics as the learning of communication, in which both domain knowledge and clear rules as well as fuzzy intuitive and context-sensitive decision making play a role. In developing agents for such problems, it is important to start with defining the functionalities that are necessary without implicitly thinking of a specific kind of implementation. By identifying what the consequences would be of implementing the different functionalities in a rule-based or a data-driven manner, insights into the preferable kind of implementation become clear. The last step is to find the right balance between the two while understanding how they interact with each other. Following such a process makes sure that there is a lot of control over the different levels of influence that both learning mechanisms as well as rules have over the final behavior of the agent.

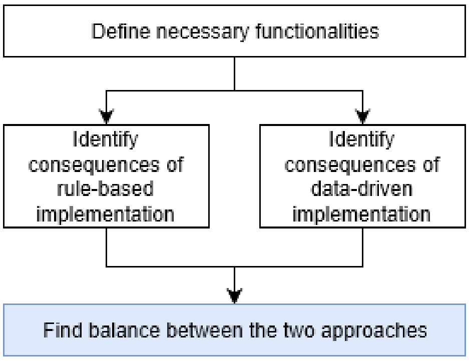

Define necessary functionalities
Identify consequences of rule-based implementation
Identify consequences of data-driven implementation
Find balance between the two approaches

Figure 24. The process for designing hybrid agents

7.1.1 Relevance of Human-Agent Experiment

In existing work on communication in human-agent teaming, either the developed agents are only evaluated in simulation (Foerster et al., 2016; Sukhbaatar et al., 2016; Unhelkar & Shah, 2016), which is mostly the case when learning is involved, or human-agent teams are only evaluated on task performance (Butchibabu, 2016; Li et al., 2016), which is mostly the case when there are set communication strategies. In the former, testing with humans is often mentioned as a point of future work. Since working together in a team is usually a long process, qualitative aspect are relevant and might influence task performance in the long run. As was shown in the results of the experiment, the type of communication that agents use clearly influence the extent to which people feel they collaborate with the agent as a team, where they prefer communication about goals. Also, if the agent communicated irrelevant information, it made them confused, causing them to check what the agent was doing. This was only one of the ways in which the humans attempted to interpret the behavior of the agents, as they constantly tried to explain the agent's behavior with behavior they would expect from a human. From this it can be concluded that qualitative evaluation through questionnaires as well as interviews gives interesting insights on the development of proactively communicating agents, that can be used to improve algorithms in the future.

Moreover, while it is often thought that letting agents learn from humans is cumbersome and will never work, it was found that agents with the right state representations can actually learn socially desirable behavior from humans after pre-training in simulation. This opens up new research opportunities for looking into the joint learning of both the human and the agent as they collaborate, using the two-phase approach of training in simulation first and with humans after.

Last, while it was found that goal-communication is more important for good qualitative evaluation of communicating agents than belief-communication, a third type of communication that is very relevant in the context of human-agent teaming emerged from the interviews, namely the communication of problems or failure. While this can be seen as a form of explicit communication as characterized by (Butchibabu, 2016), especially if it is asking for help with a problem, it could also be defined as implicit communication, especially if it were mostly about announcing a failure. It will be better to define a new type of communication for this category of messages in future work, as the communication of problems requires different kinds of relevance factors than the communication of beliefs or goals.

8. Conclusion

The aim of this thesis was to study whether it is possible to have agents learn to communicate information about observations and plans proactively in order to improve human-agent team performance, with the use of context information (RQ). As a task environment, the Blocks World for Teams environment was chosen in order to have a testbed that could be used to evaluate agents as well as human-agent teams on their collaboration and communication. A broad perspective was taken on the topic of proactive communication in teaming, by reviewing literature on human (virtual) teams, algorithms for (learning) communication, communication in human-agent teaming specifically and the evaluation of communication in teaming. This served to make sure enough attention was paid to both the development of the agents as well as the needs of the human team members.

The first sub question, **SQ1**, was about how data-driven learning methods can be balanced with rule-based methods to achieve fast learning of useful proactive communication strategies. In real-life teaming contexts, the use of proactive communication requires subtle understanding and adaptation to the needs of team members, but there are also set agreements and clear rules that determine the basis of such communication. Also, literature about algorithms for communication showed that the learning of communication is a difficult problem and that many existing systems are rule-based. Using these insights as a basis, agents were developed that make use of a reinforcement learning algorithm to learn which information should be communicated in which contexts. A BDI-program was used as a basis for the behavior of the agent and to determine the state representation as used in the learning algorithm. A process was used in which for each communication functionality and context factor, the influence of using either a rule-based or a machine learning implementation was carefully reflected upon to choose the right balance of implementation. The developed agents were able to learn proactive communication behavior that indeed improved task performance while keeping the number of messages low, within 100 to 200 simulation rounds. When qualitatively looking at the learned behavior, it made sense when attempted to explain why this should be the right communication behavior in the task. This means that the approach indeed caused useful learning outcomes.

Sub question **SQ2** dealt with the influence of the different parameters state representation, exploration and reward on the learning process and outcomes of proactive communication behavior. Agents were able to learn their proactive communication using only a negative reward each time a message was communicated and a reward based on the task performance. The height of the task performance reward influenced the number of times in which agents would decide to communicate; a higher reward appeared to favor communication. It depends on the kind of information that the agents are sharing whether it is better to have a higher or a lower reward. Sharing information about goals has a lot of influence on the task, meaning that more communication is better in many situations, and a higher reward results in a better result. Sharing observations, on the other hand, has a smaller effect on task performance; sharing a lot of information does not provide much extra value. Therefore, a lower reward improved the learning process for the communication of beliefs. Last, the effect of exploration was tested. Exploration is a difficult parameter in proactive communication sharing. In the current work agents evaluated at every timestep whether it would be a good idea to communicate certain information, meaning that this decision is made very often. Most times, the decision should be to remain silent, but a high exploration rate causes the agent to randomly decide 'communicate' too often. Using low exploration rates or a decaying exploration rate drastically brought down the number of messages and generally also improved the learning curve when combined with the proper height of the reward. Having too low of an exploration rate, however, sometimes caused a decrease in quality of the learned model.

To be able to evaluate the trained agents in a setting with humans, it was necessary to identify proper evaluation methods, as stated in **SQ3**. Different methods were extracted from the literature and used in an experiment with humans. Task performance was measured by logging the time it took for the human-agent teams to finish a task sequence. Interviews were conducted to get a view of people's qualitative understanding and evaluation of the agents. In addition they were asked to rate the extent to which they felt they had collaborated together with the agent as a team. Trust in the agent and usability of the agent's communication were evaluated using a questionnaire. Both the subjective team experience score and the usability value yielded significant results, meaning that these measures were able to capture at least some of the differences between the conditions. For trust, however, there was no significant difference, and a lot of variety in answers between the different questions about trust could be seen. This shows that the questions used were probably not the right way to measure trust in the current human-agent teaming context.

In the mentioned experiment, it was evaluated how information sharing of observations and plans by both humans and agents influences team performance (**SQ4**) and how well the learned behavior of the simulation trained agents would transfer to a context with humans (**SQ5**). As mentioned above, a significant effect was found for subjective team experience and usability, where agents that communicate their goals (plans) score higher on both measures than agents that communicate only information about their environment (observations). In a condition that combined communication about the environment with communication about goals, qualitative analysis of interviews showed that some participants perceived the agent as communicating truly proactively. It was mentioned that the agents communicated exactly the information that the human needed several times. Looking at the learned model of the agent, it could be seen that while some learned behaviors had transferred from simulation runs, indeed other new rules that more clearly and precisely defined proactive behavior had been added to the learning model.

The outcomes of this research project serve as a guide for how proactively communicating agents might be developed, and the different aspects and problems that have to be taken into account. It can be concluded that it is possible to have an agent learn how to communicate proactively in a team setting. Also, learned behavior can be transferred from a simulation context to a context where agents learn with humans, where the agents continue to adapt their behavior to those humans. It therefore paves the path for future research into the development of agents using more complex learning algorithms in more complex environments than were used in this study, while keeping the needs of the human team members in mind.

9. Future Work

Since the work done is of an exploratory nature, there are several aspects that pose challenges for future research in proactive communication for human-agent teaming. The opportunities described below should therefore mostly be seen as a challenge and inspiration for other researchers to continue work on proactive communication.

- **BW4T**: The environment used in this study, Blocks World 4 Teams, is a simplified and limited environment. While this makes it easy to program agents and let them learn some behavior, it simplifies the problem, and it is hard to determine how well this scales to real world scenarios. In future work, it will be useful to look into more complicated environments, to see if learning mechanisms such as designed in this study still work in such environments. Especially factors such as adding stochasticity in the task or environment might give some interesting insights. Also, it would be relevant to look into the use of dependencies and different capabilities of the team members, to test what the influence of those is on communication strategies.

- **Reinforcement Learning:** For the learning mechanism of the agents, a choice was made for Q-learning. However, Q-learning is a relatively simple algorithm that does not scale well to large state spaces. This problem was partly bypassed by simplifying the state space, showing that even with a simple algorithm such as Q-learning, it is possible to learn proactive communication. In future research, it will be relevant to look into more complex algorithms, especially those that are able to generalize over states such as Deep Q-Networks (Mnih et al., 2015). Using a Neural Network enables the agent to effectively learn what the influence of the different state (or relevance) factors are on a communication decision. In a new situation, an agent would then be able to simply analyze the values for all state factors and calculate the best communication decision.

- **(De)coupling of Actions and Communication**: In order to focus on researching the learning of communication, the agents developed in this research had completely separate action selection and communication decision making, where the action selection was purely rule-based. However, already in defining the state representation for learning how to communicate goals, sometimes it was necessary to combine communication acts with actual actions to allow for true collaborative skills. It might therefore make sense to do more research in the coupling of actions and communications in partly learning systems, and how the different aspects interact with each other. Learning an action strategy next to a communication strategy might cause new complications of the problem, where both parts need to incorporate information about the other in their state representation, which makes for an interesting research challenge.

- **Determining Relevance**: As mentioned above, while currently the state or relevance factors have been determined by knowledge rules, they might also be determined by more complex algorithms or models. Especially if the agents should base their communication decision on a broader definition of context, it will be interesting to look into the integration of more complex models into the communication learning architecture. Future research might for example attempt to integrate a model of the user's workload or a risk attached to not communicating something to the state representation of the learning mechanism. While this will take more computing power, it will be interesting to see if using a BDI-based agent can indeed help to simplify output of such models to the right extent, such that the learning of reasonable behavior is still possible within a small amount of simulation runs.

- **Experiment**: In the experiment with humans, several choices were made that might have influenced the results. First of all, a clear distinction was made between the communication of goals and beliefs, since previous research found differences between the two and also because both require a different way of looking at the relevance of a message. However, in the third condition in the experiment, both were combined, seemingly yielding better communication behavior than both separate versions. The interaction between the two types of communication existing next to each other might influence what is preferable behavior. In future research it will be interesting to look at agents that learn more complete communication strategies at once, to especially see if agents combine the communication of certain information as well. For example, agents already learned to communicate to which room it was going, and to also communicate that the next color block was present in that room, basically generating an explanation for its actions. It would be interesting to see if there can be more emerging patterns like those.

Last, the qualitative evaluations of the agents might be slightly biased towards a positive performance of the goal-communicating agent, simply because of the nature of the BW4T task. In this context, since both human as well as agent team members are constantly engaged with the task, information about goals or actions is basically always relevant, as it is generally useful to know what the other is doing. In a task or context in which both team members also have individual responsibilities, or in which the responsibilities are asymmetrical, this is no longer the case. Proactively communicating goals will then also be much more difficult and prone to errors. Therefore, it would be relevant to repeat the experiment in a slightly different task, to see if people's preference for goal communication still holds.

References

Adobbati, R., Marshall, A. N., Scholer, A., Tejada, S., Kaminka, G., Schaffer, S., & Sollitto, C. (2001). Gamebots: A 3d virtual world test-bed for multi-agent research. In *Proceedings of the second international workshop on Infrastructure for Agents MAS and Scalable MAS*. https://doi.org/10.1309/AJCPH7X3NLYZPHBW

Airiau, S., Padgham, L., & Sardina, S. (n.d.). Incorporating Learning in BDI agents. In *Workshop at AAMAS 2008: Adaptive and Learning Agents* (pp. 49–56).

Allen, J. E., Guinn, C. I., & Horvtz, E. (1999). Mixed-initiative interaction. *IEEE Intelligent Systems*. https://doi.org/10.1109/5254.796083

Augello, A., Gentile, M., Weideveld, L., & Dignum, F. (2016). A model of a social chatbot. In *Smart Innovation, Systems and Technologies*. https://doi.org/10.1007/978-3-319-39345-2_57

Baarslag, T., Fujita, K., Gerding, E. H., Hindriks, K., Ito, T., Jennings, N. R., … Williams, C. R. (2013). Evaluating practical negotiating agents: Results and analysis of the 2011 international competition. *Artificial Intelligence*. https://doi.org/10.1016/j.artint.2012.09.004

Bădică, A., Bădică, C., Ganzha, M., Ivanović, M., & Paprzycki, M. (2016). Experiments with multiple BDI agents with dynamic learning capabilities. In *Communications in Computer and Information Science*. https://doi.org/10.1007/978-3-319-39387-2_23

Barto, A. G., & Mahadevan, S. (2003). Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems: Theory and Applications*. https://doi.org/10.1023/A:1022140919877

Bernsen, N. O., & Dybkjær, L. (2005). Building Usable Spoken Dialogue Systems Some Approaches. *Sprache Und Datenverarbeitung*, *28*(2).

Bordes, A., Boureau, Y.-L., & Weston, J. (2017). Learning End-To-End Goal-Oriented Dialog. In *5th International Conference on Learning Representations*.

Bordini, R. H., Hübner, J. F., & Wooldridge, M. (2007). *Programming Multi-Agent Systems in AgentSpeak using Jason*. *Programming Multi-Agent Systems in AgentSpeak using Jason*. https://doi.org/10.1002/9780470061848

Bradshaw, J. M., Sierhuis, M., Acquisti, A., Feltovich, P., Hoffman, R., Jeffers, R., … van Hoof, R. (2003). Adjustable Autonomy and Human-Agent Teamwork in Practice : An Interim Report on Space Applications. In *Agent autonomy* (pp. 9–39).

Broekens, J., Hindriks, K., & Wiggers, P. (2012). Reinforcement learning as heuristic for action-rule preferences. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-642-28939-2_2

Busetta, P., Rönnquist, R., Hodgson, A., & Lucas, A. (1999). Jack intelligent agents-components for intelligent agents in java. *AgentLink News Letter*. https://doi.org/10.1.1.30.6936

Buşoniu, L., Babuška, R., & De Schutter, B. (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*. https://doi.org/10.1109/TSMCC.2007.913919

Butchibabu, A. (2016). *Anticipatory Communication Strategies for Human Robot Team Coordination*.

Butchibabu, A., Sparano-Huiban, C., Sonenberg, L., & Shah, J. (2016). Implicit Coordination Strategies for Effective Team Communication. *Human Factors*. https://doi.org/10.1177/0018720816639712

Chai, J. Y., She, L., Fang, R., Ottarson, S., Littley, C., Liu, C., & Hanson, K. (2014). Collaborative effort towards common ground in situated human-robot dialogue. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction - HRI '14*. https://doi.org/10.1145/2559636.2559677

Chu-Carroll, J. (2000). MIMIC: An adaptive mixed initiative spoken dialogue system for information queries. *Proceedings of the Sixth Conference on Applied Natural Language Processing*. https://doi.org/10.3115/974147.974161

Costa, A. C., & Anderson, N. (2011). Measuring trust in teams: Development and validation of a multifaceted measure of formative and reflective indicators of team trust. *European Journal of Work and Organizational Psychology*. https://doi.org/10.1080/13594320903272083

Dastani, M., Mol, C., Tinnemeier, N. A. M., & Meyer, J. J. C. (2007). 2APL: A practical agent programming language. In *Belgian/Netherlands Artificial Intelligence Conference*. https://doi.org/10.1007/978-3-540-79043-3_7

De Cubber, G., Dorofteri, D., Baudoin, Y., Serrano, D., Chintamani, K., Sabino, R., … Huang, T. (2012). Desiging Intelligent robots for human-robot teaming in Urban Search and Rescue. *KI - Künstliche Intelligenz*. https://doi.org/10.1007/s13218-015-0352-5

De Jong, B. A., Dirks, K. T., & Gillespie, N. (2016). Trust and team performance: A meta-analysis of main effects, moderators, and covariates. *Journal of Applied Psychology*. https://doi.org/10.1037/apl0000110

Deljoo, A., Gommans, L., Van Engers, T., & De Laat, C. (2017). What is going on: Utility-based plan selection in BDI agents. In *AAAI Workshop - Technical Report*.

Dialogflow. (2018). Retrieved from https://dialogflow.com/

Diggelen, J. Van, Bradshaw, J. M., Grant, T., Johnson, M., & Neerincx, M. (2009). Policy-based design of human-machine collaboration in manned space missions. In *Proceedings - 2009 3rd IEEE International Conference on Space Mission Challenges for Information Technology, SMC-IT 2009*. https://doi.org/10.1109/SMC-IT.2009.52

Dignum, F., & Bex, F. (2018). Creating Dialogues Using Argumentation and Social Practices. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-77547-0_17

Dignum, V., & Dignum, F. (2015). Contextualized planning using social practices. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-25420-3_3

Dodge, J., Gane, A., Zhang, X., Bordes, A., Chopra, S., Miller, A., … Weston, J. (2015). Evaluating Prerequisite Qualities for Learning End-to-End Dialog Systems. https://doi.org/10.3382/ps.2012-02506

Dybkjær, L., & Bernsen, N. O. (2001). Usability evaluation in spoken language dialogue systems. In *Proceedings of the workshop on Evaluation for Language and Dialogue Systems -*. https://doi.org/10.3115/1118053.1118055

Fang, R., Doering, M., & Chai, J. Y. (2014). Collaborative Models for Referring Expression Generation in Situated Dialogue. In *Proceedings of the Twenty-Eigth AAAI Conference on Artificial Intelligence* (pp. 1544–1550).

Foerster, J. N., Assael, Y. M., de Freitas, N., & Whiteson, S. (2016). Learning to Communicate with Deep Multi-Agent Reinforcement Learning. *Advances in Neural Information Processing Systems*, 2137–2145.

Goddeau, D., Meng, H., Polifroni, J., Seneff, S., & Busayapongchai, S. (1996). A form-based dialogue manager for spoken language applications. *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP '96*, *2*(Icslp 96), 701–704. https://doi.org/10.1109/ICSLP.1996.607458

Goldman, C. V., & Zilberstein, S. (2003). Optimizing information exchange in cooperative multi-agent systems. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems - AAMAS '03*. https://doi.org/10.1145/860575.860598

Graves, A., Wayne, G., & Danihelka, I. (2014). Neural Turing Machines. https://doi.org/10.3389/neuro.12.006.2007

Griol, D., Hurtado, L. F., Segarra, E., & Sanchis, E. (2008). A statistical approach to spoken dialog systems design and evaluation. *Speech Communication*. https://doi.org/10.1016/j.specom.2008.04.001

Harbers, M., Jonker, C., & Van Riemsdijk, B. (2012). Enhancing team performance through effective communication. In *Proceedings of the 4th Annual Human-Agent-Robot Teamwork Workshop*.

He, J., Butler, B., & King, W. (2007). Team Cognition: Development and Evolution in Software Project Teams. *Journal of Management Information Systems*, *24*(2), 261–292. https://doi.org/10.2753/MIS0742-1222240210

Henttonen, K., & Blomqvist, K. (2005). Managing distance in a global virtual team: the evolution of trust through technology-mediated relational communication. *Strategic Change*. https://doi.org/10.1002/jsc.714

Hindriks, K. V. (2009). Programming Rational Agents in GOAL. In *Multi-agent programming: Languages, platforms and applications*. https://doi.org/10.1007/978-0-387-89299-3

Jarvenpaa, S. L., & Leidner, D. E. (1999). Communication and Trust in Global Virtual Teams. *Organization Science*. https://doi.org/10.1287/orsc.10.6.791

Johnson, M., Bradshaw, J. M., Feltovich, P. J., Jonker, C. M., Van Riemsdijk, B., & Sierhuis, M. (2011). The fundamental principle of coactive design: Interdependence must shape autonomy. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-642-21268-0_10

Johnson, M., Jonker, C., Van Riemsdijk, B., Feltovich, P. J., & Bradshaw, J. M. (2009). Joint activity testbed: Blocks World for Teams (BW4T). In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-642-10203-5_26

Johnson, S. D., Suriya, C., Yoon, S. W., Berrett, J. V., & La Fleur, J. (2002). Team development and group processes of virtual learning teams. *Computers and Education*. https://doi.org/10.1016/S0360-1315(02)00074-X

Kapetanakis, S., & Kudenko, D. (2002). Reinforcement Learning of Coordination in Cooperative Multi-Agent Systems. *Proceedings Of The National Conference On Artificial Intelligence*. https://doi.org/10.1007/978-3-540-32274-0_8

Klein, G., Woods, D. D., Bradshaw, J. M., Hoffman, R. R., & Feltovich, P. J. (2004). Ten challenges for making automation a "team player" in joint human-agent activity. *IEEE Intelligent Systems*. https://doi.org/10.1109/MIS.2004.74

Kruijff, G. J. M., Janíček, M., Keshavdas, S., Larochelle, B., Zender, H., Smets, N. J. J. M., … Sulk, M. (2014). Experience in system design for human-robot teaming in urban search and rescue. In *Springer Tracts in Advanced Robotics*. https://doi.org/10.1007/978-3-642-40686-7_8

Lazaridou, A., Peysakhovich, A., & Baroni, M. (2017). Multi-Agent Cooperation and the Emergence of (Natural) Language.

Leenders, R. T. A. J., Van Engelen, J. M. L., & Kratzer, J. (2003). Virtuality, communication, and new product team creativity: A social network perspective. *Journal of Engineering and Technology Management - JET-M*. https://doi.org/10.1016/S0923-4748(03)00005-5

Lewis, J. R. (1995). IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use. *International Journal of Human-Computer Interaction*. https://doi.org/10.1080/10447319509526110

Lewis, M., Yarats, D., Dauphin, Y. N., Parikh, D., & Batra, D. (2017). Deal or No Deal? End-to-End Learning for Negotiation Dialogues.

Li, S., Sun, W., & Miller, T. (2016). Communication in human-agent teams for tasks with joint action. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-42691-4_13

Liu, C., Lowe, R., Serban, I. V, Noseworthy, M., Charlin, L., & Pineau, J. (2017). How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation.

Logan, B. (2015). A future for agent programming. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-26184-3_1

Logan, B. (2017). Future directions in agent programming. *ALP Issue*.

Lohani, M., Stokes, C., Dashan, N., McCoy, M., Bailey, C. A., & Rivers, U. E. (2017). A framework for human-agent social systems: The role of non-technical factors in operation success. In *Advances in Intelligent Systems and Computing*. https://doi.org/10.1007/978-3-319-41959-6_12

Mioch, T., Peeters, M. M. M., & Neerincx, M. A. (2018). Improving Adaptive Human-Robot Cooperation through Work Agreements. In *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. https://doi.org/10.1111/j.1600-0838.2009.01058.x

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., … Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*. https://doi.org/10.1038/nature14236

Möller, S., Kühnel, C., Engelbrecht, K.-P., Wechsung, I., & Weiss, B. (2009). A Taxonomy of Quality of Service and Quality of Experience of Multimodal Human-Machine Interaction. In *International Workshop on Quality of Multimedia Experience, QoMEx 2009*. https://doi.org/10.1109/QOMEX.2009.5246986

Möller, S., Smeele, P., Boland, H., & Krebber, J. (2007). Evaluating spoken dialogue systems according to de-facto standards: A case study. *Computer Speech and Language*, *21*(1), 26–53.

Nasirian, F., Ahmadian, M., & Lee, O.-K. (2017). AI-Based Voice Assistant Systems: Evaluating from the Interaction and Trust Perspectives. In *23rd Americas Conference on Information Systems (AMCIS)*.

Nguyen, A., & Wobcke, W. (2006). An adaptive plan-based dialogue agent: integrating learning into a BDI architecture. *… Joint Conference on Autonomous Agents and …*. https://doi.org/http://doi.acm.org/10.1145/1160633.1160771

Nourbakhsh, I. R., Sycara, K., Koes, M., Yong, M., Lewis, M., & Burion, S. (2005). Human-robot teaming for Search and Rescue. *IEEE Pervasive Computing*. https://doi.org/10.1109/MPRV.2005.13

Panait, L., & Luke, S. (2005). Cooperative multi-agent learning: The state of the art. *Autonomous Agents and Multi-Agent Systems*. https://doi.org/10.1007/s10458-005-2631-2

Parasuraman, R., Barnes, M., Cosenzo, K., & Mulgund, S. (2007). Adaptive Automation for Human-Robot Teaming in Future Command and Control Systems. *The International C2 Journal*. https://doi.org/10.1017/CBO9781107415324.004

Rao, M., & Georgeff, P. (1995). BDI-agents: From Theory to Practice. In *Proceedings of the First International Conference on Multiagent Systems (ICMAS'95)*. https://doi.org/10.1590/S0004-282X2006000600020

Riegelsberger, J., Sasse, M. A., & McCarthy, J. D. (2003). The researcher's dilemma: Evaluating trust in computer-mediated communication. *International Journal of Human Computer Studies*. https://doi.org/10.1016/S1071-5819(03)00042-9

Ritter, A., Cherry, C., & Dolan, W. (2011). Data-driven response generation in social media. *EMNLP '11 Proceedings of the Conference on Empirical Methods in Natural Language Processing*. https://doi.org/10.1039/C5RA02289D

Schaefer, K. E., Straub, E. R., Chen, J. Y. C., Putney, J., & Evans, A. W. (2017). Communicating intent to develop shared situation awareness and engender trust in human-agent teams. *Cognitive Systems Research*. https://doi.org/10.1016/j.cogsys.2017.02.002

Serban, I. V., Sordoni, A., Bengio, Y., Courville, A., & Pineau, J. (2016). Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models. In *Thirtieth AAAI Conference on Artificial Intelligence* (pp. 3776–3783). https://doi.org/10.1007/s10544-016-0060-4

Shang, L., Lu, Z., & Li, H. (2015). Neural Responding Machine for Short-Text Conversation. https://doi.org/10.3115/v1/P15-1152

Singh, D., & Hindriks, K. v. (2013). Learning to Improve Agent Behaviours in GOAL. In *Programming Multiagent Systems* (Vol. 10, pp. 158–173). https://doi.org/10.1007/978-3-642-38700-5_10

Singh, D., Sardina, S., Padgham, L., & Airiau, S. (2010). Learning Context Conditions for BDI Plan Selection. *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1 - Volume 1*. https://doi.org/10.1137/15M1021830

Singh, D., Sardina, S., Padgham, L., & James, G. (2011). Integrating learning into a BDI agent for environments with changing dynamics. In *IJCAI International Joint Conference on Artificial Intelligence*. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-420

Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., … Dolan, B. (2015). A Neural Network Approach to Context-Sensitive Generation of Conversational Responses. https://doi.org/10.1103/PhysRevB.92.155314

Sukhbaatar, S., Szlam, A., & Fergus, R. (2016). Learning Multiagent Communication with Backpropagation. *Advances in Neural Information Processing Systems*, 2244–2252.

Sukhbaatar, S., Szlam, A., Weston, J., & Fergus, R. (2015). End-To-End Memory Networks. *Advances in Neural Information Processing Systems*, 2440–2448. https://doi.org/v5

Traum, D., Marsella, S. C., Gratch, J., Lee, J., & Hartholt, A. (2008). Multi-party, multi-issue, multi-strategy negotiation for multi-modal virtual agents. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-540-85483-8_12

Unhelkar, V. V, & Shah, J. A. (2016). ConTaCT : Deciding to Communicate during Time-Critical Collaborative Tasks in Unknown , Deterministic Domains. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*. https://doi.org/10.1.1.725.9960

van der Vecht, B., van Diggelen, J., Peeters, M., Barnhoorn, J., & van der Waa, J. (2018). Sail: A social artificial intelligence layer for human-machine teaming. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-94580-4_21

Vergunst, N. L. (2011). *BDI-based generation of robust task-oriented dialogues*.

Vinyals, O., & Le, Q. (2015). A Neural Conversational Model. https://doi.org/10.1007/978-3-319-19291-8_22

Warkentin, M., & Beranek, P. M. (1999). Training to improve virtual team communication. *Information Systems Journal*. https://doi.org/10.1046/j.1365-2575.1999.00065.x

Weston, J., Chopra, S., & Bordes, A. (2014). Memory Networks. https://doi.org/v0

White, A., Tate, A., & Rovatsos, M. (2017). Improving plan execution robustness through capability aware maintenance of plans by BDI agents. *International Journal of Agent-Oriented Software Engineering*, *5*(4), 306–335.

Williams, J. D., & Young, S. (2007). Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language*. https://doi.org/10.1016/j.csl.2006.06.008

Williams, T., Acharya, S., Schreitter, S., & Scheutz, M. (2016). Situated open world reference resolution for human-robot dialogue. In *ACM/IEEE International Conference on Human-Robot Interaction*. https://doi.org/10.1109/HRI.2016.7451767

Xuan, P., Lesser, V., & Zilberstein, S. (2001). Communication decisions in multi-agent cooperation. In *Proceedings of the fifth international conference on Autonomous agents - AGENTS '01*. https://doi.org/10.1145/375735.376469

Yan, R. (2017). &quot;Chitty-Chitty-Chat Bot&quot;: Deep Learning for Conversational AI. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*.

Young, S., Gašić, M., Thomson, B., & Williams, J. D. (2013). POMDP-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*. https://doi.org/10.1109/JPROC.2012.2225812

Young, S. J. (2000). Probabilistic methods in spoken-dialogue systems. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*. https://doi.org/10.1098/rsta.2000.0593

Appendix A

Trust & Usability Questionnaire

Participant nr:

There is a need for me to communicate to the agent to complete the task

Disagree □ Agree

There is a need for the agent to communicate to me to complete the task

Disagree □ Agree

I can effectively complete the task by communicating with the agent

Disagree □ Agree

I am able to complete the task quickly by communicating with the agent

Disagree □ Agree

I am able to efficiently complete the task by communicating with the agent

Disagree □ Agree

I feel comfortable communicating with the agent

Disagree □ Agree

The information the agent provides is easy to understand

Disagree □ Agree

The information the agent provides is effective in helping me complete the task and scenarios

Disagree □ Agree

I like communicating with the agent

Disagree □ Agree

The agent communicates in the way I expect it to

Disagree □ Agree

Overall, I am satisfied with the communication of the agent Disagree □ Agree
The agent’s communications are sufficiently informative Disagree □ Agree
The agent communicates often enough Disagree □ Agree
The agent does not communicate too often Disagree □ Agree
The agent does not lie Disagree □ Agree
The agent’s communications are relevant Disagree □ Agree
If I had my way, I wouldn’t let the agent have influence over the completion of the task Disagree □ Agree
I would be comfortable giving the agent complete responsibility for the completion of the task Disagree □ Agree
I really wish I had a good way to oversee the work of the agent on the task Disagree □ Agree
I can rely on the agent Disagree □ Agree
Overall, the agent is very trustworthy Disagree □ Agree
I have confidence in this team

Disagree □ Agree

Team members in this team are truthful to each other

Disagree □ Agree

The agent tried to get the upper hand

Disagree □ Agree

In this team we work in a climate of cooperation

Disagree □ Agree

The agent holds back relevant information

Disagree □ Agree

In this team we provide each other with timely information

Disagree □ Agree

Appendix B

Important Values

Communication	15		
Live up to expectations	5		
Listening	4		
Understanding each other	4		
Trust	4		
Equality	4		
Motivation	3		
Respect	3	Respect agreements	1
Openness	2		
Knowing capabilities	2	Use capabilities	2
Agree about plans and goals	2	Establish common goal	1
Honesty	2	Honest	1
Division of roles/tasks	2	Agree on task divisions	1
Knowing the goal	1		
Efficiency	1		
Timing actions	1		
Explain values to each other	1	Your values	1
Helping each other	1		
Ask questions	1	Ask questions	1
Communicate on the same level	1	Communicate on the same lev	1
Punctuality	1		
Constructive attitude	1		
Possibility to express opinion	1	Possibility for discussion	1
Equal vision	1		
Don't be too stubborn	1		
Social bond	1	Social/Getting to know each c	1
Knowing capabilities	1		
Reliability	1		
Supportive attitude	1	Constructive	1

What you're doing (goals and plans)	14
Possible problems	9
Clear	8
Transparent	4
Timely	4
Frequent	3
Expectations	2
Use capabilities	2
Logical	1
Respect agreements	1
Not too much	1
Balance communication and individual effort	1
Instructions	1
Establish common goal	1
Agree on task divisions	1
Calmly	1
Patient (wanting to repeat if necessary)	1
Ask questions	1
Communicate on the same level	1
Take feelings into account	1
Planning meetings	1
If you don't like something	1
Your values	1
Social/Getting to know each other	1
Feedback	1
Visual	1
Through speech	1
Possibility for discussion	1
React to each other	1
Constructive	1
Understandable	1
Right amount of information	1

Communication	15
Live up to expectations	5
Listening	4
Understanding each other	4
Trust	4
Equality	4
Respect	4
Knowing capabilities	4
Motivation	3
Agree about plans and goals	3
Honesty	3
Division of roles/tasks	3
Openness	2
Explain values to each other	2
Ask questions	2
Communicate on the same level	2
Possibility to express opinion	2
Social bond	2
Supportive attitude	2
Knowing the goal	1
Efficiency	1
Timing actions	1
Helping each other	1
Punctuality	1
Constructive attitude	1
Equal vision	1
Don't be too stubborn	1
Knowing capabilities	1
Reliability	1

Honest	1

Understood Agent Behavior

Condition 1

Round 1

Agent communicates everything in a room	5
Agent goes by rooms one by one	5
Agent is faster than human	4
Agent starts at bottom left room	4
Agent does not anticipate	3
Agent goes for first block	3
Agent works on its own	1
Unsure about anticipation of agent	1
Agent starts at bottom rooms	1
Agent repeats observations	1
Agent doesn't listen	1
Agent understands bottom rooms are close	1
Agent gives incorrect information	1
Unsure if agent communicates all observations	1
Agent tries to communicate relevant info	1
Agent communicates one block per room	1
Agent communicates randomly	1
Agent responded to info about next block	1
Agent goes to a block the human communicates if next block	1
Agent doesn't see what the human is doing	1
Agent communicates necessary blocks	1

Round 2

Agent communicates irrelevant information	2
Agent gives incorrect information	2
Agent is 'faster'	1
Thinking agent communicates when human carries a block	1
Agent goes to a block the human communicates if next block	1

Condition 2

Round 1

Agent tells where he is going	6
Agent skips block human is doing	5
Agent does not communicate everything	4
Agent is faster than human	2
Agent communicates when he found a block	2
Agent goes to bottom rooms	2
Agent decides who does what	1
Agent wants to take turns	1
Agent often changed its mind	1
Agent goes to bottom left first	1
Agent waited for human to deliver commited block	1
Agent drops carried block when human commits	1
Agent does not go to room human committed to	1
Sometimes the agent freezes	1
Agent remembers blocks it has seen	1
Agent communicates when picking up a block	1
Agent goes to closest room first	1
Agent did not often communicate block from room	1
Agent goes for first block	1

Round 2

Agent skips block human said they're doing	1
Agent communicates more than first	1
Agent's behavior was less random	1
Agent goes to bottom left room first	1
Agent knows where a block is when he says he will get it	1

Condition 3

Round 1

Agent skips block human committed to	7
Agent communicates irrelevant information	6
Agent communicates all blocks he finds in a room	5
Agent goes to left first	4
Agent communicates relevant information	3
Agent communicates where he is going	3
Agent communicates what block he will get	1
Agent tells location of block that human commits to	1
Agent communicates faster	1
Agent pays attention to human	1
Agent communicates randomly	1
Agent goes to rooms in a sequence	1
Agent picks up block if he finds it	1
Agent comes with next block right after human dropped previous	1
Agent explored more rooms in the beginning	1
Agent went to rooms the human had already been in	1
Agent takes into account communicated information	1
Agent goes to room with necessary block if communicated by human	1
Agent sometimes communicates what he finds	1

Round 2

Agent drops blocks that are not yet necessary	2
Agent communicated random block when not seeing useful blocks	1
Agent goes to dropzone without dropping anything	1
Agent follows the same pattern	1
Agent communicated more truthfully	1
Agent communicated more relevant information	1
Agent does not wait for human to deliver block	1
Agent tells location of block that human commits to	1

Strategies

Condition 1

Round 1

Searching for second block	6
Searching for third block	3
Starting at top rooms	2
Start at the bottom right	2
Searching for blocks in bottom rooms	2
Remember agent communications	1
Paying attention to agent communication	1
Remember where blocks are	1
Communicating relevant blocks while walking	1
Taking turns with blocks	1
Anticipate speed of agent	1
Starting with middle rooms	1
Go for block one if it's closeby	1

Round 2

Communicate relevant blocks	5
Trying to communicate better	2
Starting with top rooms	2
Searching for third block	2
Looking further ahead	1
Not picking up blocks	1
Not remembering blocks	1
Ignoring agent's communication	1
Exploring rooms	1
Relying on agent for picking up blocks	1
Letting the agent check bottom rooms	1
Starting with middle rooms	1
Communicating as much as possible	1
Communicating blocks further in the sequence	1
Letting the agent sometimes do something	1
Searching for second block	1
Communicate first two blocks	1
Trying to mislead agent	1
Letting the agent do bottom rooms	1
Pay attention to location of agent	1
Pay attention to agent's communication	1

Condition 2

Round 1

Searching for the second block	4
Taking turns with bottom rooms	1
Not going too far away	1
Searching for the third block	1
Relying on agent for first blocks	1
Tell agent which block they will bring	1
Explore rooms	1
Pay attention to agent's communication	1
Estimate agent's behavior	1
Communicate as much as possible	1
Taking turns with blocks	1
Going to the closest room	1
Not following the agent	1
Look forward	1
Pick up the first block if accidentally found	1
Starting with the top rooms	1
Communicate where they are going	1
Take responsibility for one row of rooms	1
Starting with the bottom rooms	1
Communicate which block they are getting	1

Round 2

Do not communicate color when two in a row	2
Searching in bottom rooms	1
Remember location of blocks	1
Search for next blocks while carrying block	1
Start with second block	1
Communicate more	1
Only communicate when they found a block	1
Pay more attention to agent after communication	1
Looking further ahead	1
Searching for the third block	1
Communicating less often	1
Do not communicate last block	1

Condition 3

Round 1		**Round 2**	
Skipping block that agent commits to	3	Starting with middle rooms	3
Starting at the right	3	Communicate more	3
Paying attention to agent communication about goals	2	Searching for the second block	2
Taking turns in delivering blocks	2	Telling what they are going to do	1
Go the opposite way as the agent	2	Not trying to remember agent's messages	1
Communicate everything they do	2	Do not communicate color when two in a row	1
Communicate location of blocks	2	Communicate where they are going	1
Waiting for who finds the first block	1	Looking where the agent is going	1
Searching for the second block	1	Pay more attention to agent's behavior	1
Starting at the top rooms	1	Relying more on the agent	1
Pay attention to agent communication about blocks	1	Communicate what is in a room while walking	1
Starting at the bottom rooms	1		

Evaluation of Communication

Condition 1

Communication disappears too quickly	6
Unequal roles	3
Communication is confusing	3
Able to predict agent behavior	3
Not knowing what the agent is doing	3
Communication is useful	2
Communication is fine	2
Feeling like the agent takes the lead	1
Understanding agent after explanation	1
Feeling less smart than the agent	1
Working together effectively	1
Only small improvements necessary	1
Communication is quite smooth	1
Knowing what to expect of the agent	1
Not feeling like the agent was listening	1
Too much communication	1
Unclear communication	1
One-sided communication	1
Agent tries his best	1
Agent's information can be trusted	1
Communication is not effective	1
Agent listens	1
Communication is limited	1

Condition 2

Trusting the agent	4
Communication disappears	4
Communication is useful	3
Not knowing where the agent is	2
Missing a personal part	2
Not enough communication	2
Possible to predict agent behavior	2
Clear communication	2
Agent is transparent	2
Communication is limited	2
Communication is annoying	1
Agent's communication is clear	1
Agent's communication is direct	1
Unclear communication	1
Get comfortable with agent communication over time	1
One-sided communication	1
Communication was not always good	1
Wanting to check the agent	1
Unequal roles	1

Condition 3

Too many messages	4
Communication disappears	3
Communication is useful	3
Communication is confusing	2
Knowing what the agent would do	2
Trusting in the agent	2
Clear communication	2
Agent did well as a team member	1
Not enough communication	1
Agent was supportive	1
Unclear communication	1
Communication is annoying	1
Agent is helpful	1
Agent is efficient	1

Possible Improvements

Condition 1

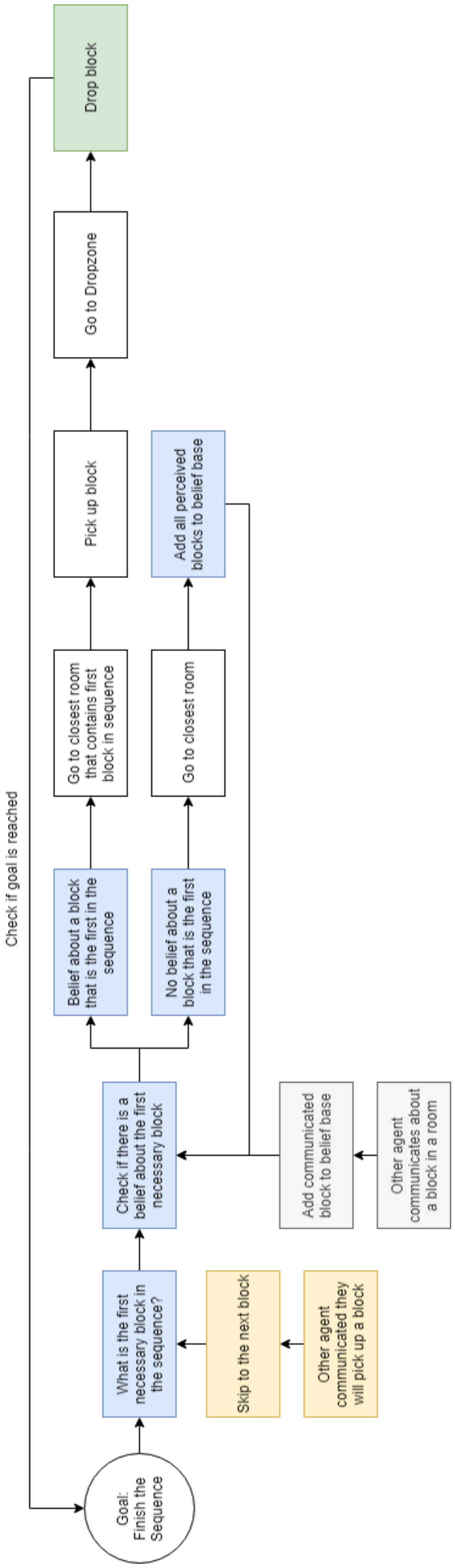
Communicate relevant information only	6
Agent should communicate plans	5
Communications should not disappear	3
Overview of messages	3
Using visuals instead of text messages	2
Less communication	1
More equal communication	1
Agent should anticipate to team member	1
Clicking can be more efficient	1
Ability to discuss task division	1
Possibility to do suggestions	1
Possibility to ask questions	1
Possibility to plan beforehand	1
More communication options	1

Condition 2

Ability to discuss task division	4
More communication	3
Communicate information about blocks	3
Separate communication from actions	2
Having an image of the agent	1
More communication options	1
Type your own messages	1
Using visuals instead of text messages	1
Possibility of confirmation of finding a block	1
Possibility of telling the agent what not to do	1
Ability to look back in messages	1
Possibility to say nothing relevant is in a room	1
More human-like communication	1
More social communication	1

Condition 3	
Communicate relevant information only	4
Overview of messages	2
Communications should not disappear	2
Division between messages from different team members	1
Ability to discuss strategy	1
Repetition of relevant information	1
Possibility to communicate when something goes wrong	1
Communicate earlier	1

Appendix C


Goal: Finish the Sequence
What is the first necessary block in the sequence?
Skip to the next block
Other agent communicated they will pick up a block
Check if there is a belief about the first necessary block
Add communicated block to belief base
Other agent communicates about a block in a room
Belief about a block that is the first in the sequence
No belief about a block that is the first in the sequence
Go to closest room that contains first block in sequence
Go to closest room
Pick up block
Add all perceived blocks to belief base
Go to Dropzone
Drop block
Check if goal is reached