

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without the author's prior consent.



**UNIVERSITY OF
PLYMOUTH**

**TEACHING ROBOTS SOCIAL AUTONOMY FROM
IN SITU HUMAN SUPERVISION**

by

EMMANUEL SENFT

A thesis submitted to the University of Plymouth
in partial fulfilment for the degree of

DOCTOR OF PHILOSOPHY

School of Computing, Electronics and Mathematics

2018

Acknowledgements

First of all, none of this would have been possible without Tony Belpaeme. Thank you Tony for giving me the opportunity to start this journey and meet so many people, for introducing me to research, for your enthusiasm, and for making me leave each meeting more motivated than when starting. Thank you also Paul Baxter, for your quiet but sharp comments and ideas. To Séverin Lemaignan for your energy and positivity, and your craving for technical complexity. Thank you to the three of you, for guiding me through these 4 years.

Special thanks to James Kennedy for showing me what a PhD journey could be, for these 2 years and half sharing the office, and for an ‘interesting’ week in Romania to make DREAM work. Thank you to the DREAM team, for the fun of sharing meetings and the opportunity to collaborate on a bigger project, and especially to Hoang-Long Cao and Pablo Gomez for our weekly meetings. To Carlos Cifuentes, Marcela Munera and Jonathan Casas for a warm welcome in Colombia and a productive collaboration. To Madeleine Bartlett for long weeks in schools, repeating the same sentences to more than 100 children and teaching the robot. Also thanks to Charlotte Edmunds, Madeleine Bartlett, Chris Wallbridge, Daniel Hernandez, and Thomas Colin for accepting to proofread the whole thing, dealing with the frenchness of my spelling and grammar, making it definitely better! To everybody not mentioned yet and who has been part of the HRI Plymouth team: Robin Read, Bahar Irfan, Fotios Popadopoulos, and Serge Thill; to the Cognovians for the warm welcome on my arrival in Plymouth and people in CRNS for being a fun and vibrant research group.

The last thoughts are for the ones who gave me the energy to enjoy working on this thesis: people in North Road East and in Arundel Crescent who made the house a pleasant and welcoming place I was always happy to return in the evenings, and friends in Plymouth and around the world (Pierre-Henri, Thomas and Tunvez to cite only a few) who provided me with pleasant distractions.

Final thanks to my family and Yan for supporting me through the highs and lows and making me feel appreciated at all times.

Author's declaration

At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award without prior agreement of the Doctoral College Quality Sub-Committee.

Work submitted for this research degree at the University of Plymouth has not formed part of any other degree either at the University of Plymouth or at another establishment.

This work has been carried out by Emmanuel Senft under the supervision of Prof. Dr. Tony Belpaeme, Dr. Paul Baxter, and Dr. Séverin Lemaignan. The work was funded by European Union FP7 projects DREAM (grant no.: 611391).

Parts of this thesis have been published by the author:

Senft, E., Baxter, P., & Belpaeme, T. (2015a). Human-Guided Learning of Social Action Selection for Robot-Assisted Therapy. In *Proceedings of the Workshop on Machine Learning for Interactive Systems (MLIS'15)*, at ICML, (pp. 15–20). JMLR Workshop & Conference Series

Senft, E., Baxter, P., Kennedy, J., & Belpaeme, T. (2015b). SPARC: Supervised Progressively Autonomous Robot Competencies. In *Proceedings of the International Conference on Social Robotics (ICSR)*, (pp. 603–612). Springer. https://doi.org/10.1007/978-3-319-25554-5_60

Senft, E., Baxter, P., Kennedy, J., & Belpaeme, T. (2015c). When is It Better to Give Up?: Towards Autonomous Action Selection for Robot Assisted ASD Therapy. In *Proceedings of the 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI) Extended Abstracts*, (pp. 197–198). ACM. <https://doi.org/10.1145/2701973.2702715>

Senft, E., Baxter, P., Kennedy, J., Lemaignan, S., & Belpaeme, T. (2016a). Providing a Robot With Learning Abilities Improves Its Perception by Users. In *Proceedings of the 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI) Extended Abstracts*, (pp. 513–514). IEEE Press. <https://doi.org/10.1109/HRI.2016.7451832>

Senft, E., Lemaignan, S., Baxter, P. E., & Belpaeme, T. (2016b). SPARC: an Efficient Way to Combine Reinforcement Learning and Supervised Autonomy. In *Proceedings of the Future of Interactive Learning Machines Workshop (FILM)*, at NIPS

Senft, E., Baxter, P., Kennedy, J., Lemaignan, S., & Belpaeme, T. (2017a). Supervised Autonomy for Online Learning in Human-Robot Interaction. *Pattern Recognition Letters*, 99, 77–86. <https://doi.org/10.1016/j.patrec.2017.03.015>

Senft, E., Lemaignan, S., Baxter, P., & Belpaeme, T. (2017b). Toward Supervised Reinforcement Learning With Partial States for Social HRI. In *Proceedings of the*

Artificial Intelligence for Human-Robot Interaction Symposium (AI-HRI), at AAAI Fall Symposium Series.

Senft, E., Lemaignan, S., Baxter, P. E., & Belpaeme, T. (2017c). Leveraging Human Inputs in Interactive Machine Learning for Human Robot Interaction. In *Proceedings of the Companion of the 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 281–282). ACM. <https://doi.org/10.1145/3029798.3038385>

Senft, E., Lemaignan, S., Bartlett, M., Baxter, P., & Belpaeme, T. (2018a). Robots in the Classroom: Learning to Be a Good Tutor. In *Proceedings of the 4th Workshop on Robots for Learning (R4L) - Inclusive Learning, at HRI*

Senft, E., Lemaignan, S., Baxter, P., & Belpaeme, T. (2018b). From Evaluating to Teaching: Rewards and Challenges of Human Control for Learning Robots . In *Proceedings of the Workshop on Human/Robot in the Loop Machine Learning (HRML), at IROS*

Collaborative work published but not included in this research:

Kennedy, J., Baxter, P., Senft, E., & Belpaeme, T. (2015b). Higher Nonverbal Immediacy Leads to Greater Learning Gains in Child-Robot Tutoring Interactions. In *Proceedings of the International Conference on Social Robotics (ICSR)*, (pp. 327–336). Springer International Publishing

Baxter, P., Matu, S., Senft, E., Costescu, C., Kennedy, J., David, D., & Belpaeme, T. (2015b). Touchscreen-Mediated Child-Robot Interactions Applied to ASD Therapy. In *Proceedings of the 1st International Conference on Social Robots in Therapy and Education*. Almere, Netherlands

Kennedy, J., Baxter, P., Senft, E., & Belpaeme, T. (2015c). Using Immediacy to Characterise Robot Social Behaviour in Child-Robot Interactions. In *Proceedings of the 1st Workshop on Evaluating Child-Robot Interaction, at ICSR*

Baxter, P., Ashurst, E., Kennedy, J., Senft, E., Lemaignan, S., & Belpaeme, T. (2015a). The Wider Supportive Role of Social Robots in the Classroom for Teachers. In *Proceedings of the 1st International Workshop on Educational Robotics, at ICSR*

Kennedy, J., Baxter, P., Senft, E., & Belpaeme, T. (2016b). Social Robot Tutoring for Child Second Language Learning. In *Proceedings of the 11th ACM/IEEE International Conference on Human Robot Interation (HRI)*, (pp. 231–238). IEEE Press

Baxter, P., Kennedy, J., Senft, E., Lemaignan, S., & Belpaeme, T. (2016). From Characterising Three Years of HRI to Methodology and Reporting Recommendations. In *Proceedings of the 11th ACM/IEEE International Conference on Human Robot Interaction (HRI)*, (pp. 391–398). IEEE Press

Kennedy, J., Baxter, P., Senft, E., & Belpaeme, T. (2016a). Heart vs Hard Drive: Children Learn More From a Human Tutor Than a Social Robot. In *Proceedings of the 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI) Extended Abstracts*, (pp. 451–452). IEEE

Wills, P., Baxter, P., Kennedy, J., Senft, E., & Belpaeme, T. (2016). Socially Contingent Humanoid Robot Head Behaviour Results in Increased Charity Donations. In *Pro-*

ceedings of the 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI) Extended Abstracts, (pp. 533–534). IEEE

Kennedy, J., Lemaignan, S., Montassier, C., Lavalade, P., Irfan, B., Papadopoulos, F., Senft, E., & Belpaeme, T. (2017). Child Speech Recognition in Human-Robot Interaction: Evaluations and Recommendations. In *Proceedings of the 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 82–90). ACM

Esteban, P. G., Baxter, P., Belpaeme, T., Billing, E., Cai, H., Cao, H.-L., Coeckelbergh, M., Costescu, C., David, D., De Beir, A., et al. (2017). How to Build a Supervised Autonomous System for Robot-Enhanced Therapy for Children With Autism Spectrum Disorder. *Paladyn, Journal of Behavioral Robotics*, 8(1), 18–38

Lara, J. S., Casas, J., Aguirre, A., Munera, M., Rincon-Roncancio, M., Irfan, B., Senft, E., Belpaeme, T., & Cifuentes, C. A. (2017). Human-Robot Sensor Interface for Cardiac Rehabilitation. In *Proceedings of the International Conference on Rehabilitation Robotics (ICORR)*, (pp. 1013–1018). IEEE

Lemaignan, S., Edmunds, C. E. R., Senft, E., & Belpaeme, T. (2018). The PInSoRo Dataset: Supporting the Data-Driven Study of Child-Child and Child-Robot Social Dynamics. *PLOS ONE*, 13(10), 1–19.

URL [Https://doi.org/10.1371/journal.pone.0205999](https://doi.org/10.1371/journal.pone.0205999)

Irfan, B., Kennedy, J., Lemaignan, S., Papadopoulos, F., Senft, E., & Belpaeme, T. (2018). Social Psychology and Human-Robot Interaction: an Uneasy Marriage. In *Companion of the 13th ACM/IEEE International Conference on Human-Robot Interaction (Alt.HRI)*

Cao, H.-L., Van de Perre, G., Kennedy, J., Senft, E., Esteban, P. G., De Beir, A., Simut, R., Belpaeme, T., Lefeber, D., & Vanderborght, B. (2018). A Personalized and Platform-Independent Behavior Control System for Social Robots in Therapy: Development and Applications. *IEEE Transactions on Cognitive and Developmental Systems*

Casas, J., Irfan, B., Senft, E., Gutiérrez, L., Rincon-Roncancio, M., Munera, M., Belpaeme, T., & Cifuentes, C. A. (2018). Social Assistive Robot for Cardiac Rehabilitation: a Pilot Study With Patients With Angioplasty. In *Companion of the 13th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 79–80). ACM

Wallbridge, C. D., Lemaignan, S., Senft, E., Edmunds, C., & Belpaeme, T. (2018). Spatial Referring Expressions in Child-Robot Interaction: Let's Be Ambiguous! In *Proceedings of the 4th Workshop on Robots for Learning (R4L) - Inclusive Learning, at HRI*

Word count for the main body of this thesis: **57,810**

Signed: _____

Date: _____

Abstract

TEACHING ROBOTS SOCIAL AUTONOMY FROM IN SITU HUMAN SUPERVISION

Emmanuel Senft

Traditionally the behaviour of social robots has been programmed. However, increasingly there has been a focus on letting robots learn their behaviour to some extent from example or through trial and error. This on the one hand excludes the need for programming, but also allows the robot to adapt to circumstances not foreseen at the time of programming. One such occasion is when the user wants to tailor or fully specify the robot's behaviour. The engineer often has limited knowledge of what the user wants or what the deployment circumstances specifically require. Instead, the user does know what is expected from the robot and consequently, the social robot should be equipped with a mechanism to learn from its user.

This work explores how a social robot can learn to interact meaningfully with people in an efficient and safe way by learning from supervision by a human teacher in control of the robot's behaviour. To this end we propose a new machine learning framework called Supervised Progressively Autonomous Robot Competencies (SPARC). SPARC enables non-technical users to control and teach a robot, and we evaluate its effectiveness in Human-Robot Interaction (HRI). The core idea is that the user initially remotely operates the robot, while an algorithm associates actions to states and gradually learns. Over time, the robot takes over the control from the user while still giving the user oversight of the robot's behaviour by ensuring that every action executed by the robot has been actively or passively approved by the user. This is particularly important in HRI, as interacting with people, and especially vulnerable users, is a complex and multidimensional problem, and any errors by the robot may have negative consequences for the people involved in the interaction.

Through the development and evaluation of SPARC, this work contributes to both HRI and Interactive Machine Learning, especially on how autonomous agents, such as social robots, can learn from people and how this specific teacher-robot interaction impacts the learning process. We showed that a supervised robot learning from their user can reduce the workload of this person, and that providing the user with the opportunity to control the robot's behaviour substantially improves the teaching process. Finally, this work also demonstrated that a robot supervised by a user could learn rich social behaviours in the real world, in a large multidimensional and multimodal sensitive environment, as a robot learned quickly (25 interactions of 4 sessions during in average 1.9 minutes) to tutor children in an educational game, achieving similar behaviours and educational outcomes compared to a robot fully controlled by the user, both providing 10 to 30% improvement in game metrics compared to a passive robot.

Contents

Acknowledgements

Author's declaration

Abstract

1	Introduction	1
1.1	Scope	2
1.1.1	Frame	2
1.1.2	Type of Interaction	4
1.2	The Thesis	5
1.3	Research Approach	7
1.3.1	Review of the State of the Art in HRI	7
1.3.2	New Teaching Framework	7
1.3.3	Study Design	8
1.3.4	Learning Algorithms	9
1.3.5	Terminology	10
1.4	Contributions	10
1.4.1	Scientific Contributions	11
1.4.2	Technical Contributions	12
1.4.3	Other Technical Contributions	12
1.5	Structure	12
2	Background	15
2.1	Social Human-Robot Interaction	15
2.1.1	Fields of Application	15
2.1.2	Requirements on Robots Interacting with Humans	22
2.2	Current Robot Controllers in HRI	28
2.2.1	Scripted Behaviour	29
2.2.2	Adaptive Preprogrammed Behaviour	29
2.2.3	Wizard of Oz	31

2.2.4 Learning from Demonstration	32
2.2.5 Planning	35
2.2.6 Summary	36
2.3 Interactive Machine Learning	38
2.3.1 Goal	39
2.3.2 Active Learning	40
2.3.3 Reinforcement Learning	41
2.3.4 Human as a Source of Feedback on Actions	44
2.3.5 Interactive Learning from Demonstration	47
2.3.6 Importance of Control	48
2.4 Summary	49
3 Supervised Progressive Autonomous Robot Competencies SPARC	51
3.1 Motivation	52
3.2 Context	52
3.3 Principles	53
3.4 Goal	56
3.5 Implications	57
3.5.1 Relation with Time	57
3.5.2 Difference with Learning from Demonstration (LfD)	58
3.5.3 Interaction with Learning Algorithms	59
3.6 Application to a New Task	60
3.7 Summary	60
4 Study 1: Comparison Between SPARC and WoZ	63
4.1 Motivation	64
4.2 Scope of the Study	65
4.3 Methodology	66
4.3.1 Participants	66
4.3.2 Task	67
4.3.3 Child Model	68
4.3.4 Wizarded-Robot Control	69
4.3.5 Learning Algorithm	71

CONTENTS

4.3.6 Interaction Protocol	72
4.3.7 Metrics	73
4.4 Results	75
4.4.1 Interaction Data	75
4.4.2 Questionnaire Data	77
4.5 Discussion	78
4.6 Summary	80
5 Study 2: Importance of Control Over the Learner	81
5.1 Motivation	82
5.2 Scope of the Study	82
5.2.1 Interactive Reinforcement Learning	82
5.2.2 SPARC	83
5.2.3 Differences Between IRL and SPARC	84
5.2.4 Hypotheses	84
5.3 Methodology	85
5.3.1 Participants	85
5.3.2 Task	86
5.3.3 Implementation	87
5.3.4 Interaction Protocol	90
5.3.5 Metrics	92
5.4 Results	93
5.4.1 Interaction Data	94
5.4.2 Questionnaire Data	99
5.4.3 Expert	99
5.5 Validation of the Hypotheses	101
5.5.1 Effectiveness and Efficiency with Non-Experts	101
5.5.2 Safety with Experts	102
5.5.3 Control	102
5.6 Discussion	104
5.6.1 Comparison with Original Interactive Reinforcement Learning Study	104
5.6.2 Advantages and Limitations of SPARC	105
5.6.3 Limitations of the Evaluation	106

5.7 Summary	107
6 Study 3: Application of SPARC to Tutoring	109
6.1 Motivation	111
6.2 Scope of the Study	111
6.3 Apparatus	112
6.3.1 Food Chain Game	113
6.3.2 Test	114
6.3.3 Robot Behaviour	114
6.3.4 Control Architecture	116
6.3.5 Environment Model	119
6.3.6 Learning Algorithm	123
6.3.7 Teacher Interface	126
6.4 Methodology	129
6.4.1 Participants	129
6.4.2 Teacher	129
6.4.3 Conditions	130
6.4.4 Protocol	131
6.4.5 Metrics	132
6.5 Results	135
6.5.1 Example of a Session	135
6.5.2 Comparison of Policy	137
6.5.3 Test Performance	137
6.5.4 Game Metrics	138
6.5.5 Teaching the Robot	141
6.6 Discussion	144
6.6.1 Impacts	145
6.6.2 Limitations of the Study	149
6.6.3 Considerations When Applying SPARC	155
6.7 Summary	161
7 Discussion	163
7.1 Research Questions	164
7.2 Limitations of SPARC	167

CONTENTS

7.2.1	Requirement of a Human in the Loop	167
7.2.2	Reliance on Human's Attention	168
7.2.3	Time Pressure	169
7.2.4	Overloading the Teacher	170
7.2.5	Interface	170
7.3	Impact	171
7.3.1	Pushing the State of the Art	171
7.3.2	Empowering Non-Experts in Robotics	172
7.3.3	Robots and Proactivity	173
7.4	Future Work	174
7.4.1	Application Domains	174
7.4.2	Learning Beyond Imitation	175
7.4.3	Sustained Learning	176
7.4.4	Interface With the Teacher	177
7.4.5	Algorithms	178
7.5	Summary	179
8	Contribution and Conclusion	181
8.1	Summary	181
8.2	Contributions	183
8.3	Conclusion	184
Glossary		187
Acronyms		189
Appendices		191
A Contribution to the DREAM Project		193
A.1	Software Development	193
A.2	Tools	194
A.2.1	yarpGenerator	194
A.2.2	scriptManager	194
B yarpGenerator Technical Report		195

CONTENTS

C Social Assistive Robot for Cardiac Rehabilitation	205
D Information and Questionnaires for the First Study	207
E Information and Questionnaires for the Second Study	215
F Teacher's Diary	227
Bibliography	235

List of Figures

1.1	The typical interaction used in this research: a robot interacts with a target in an application interaction and learns from a domain expert through a teaching interaction.	4
3.1	Frame of the interaction: a robot interacts with a target, suggests actions and receive commands and feedback from a teacher. Using machine learning, the robot improves its suggestions over time, to reach an appropriate policy.	53
3.2	Flowchart of the action selection loop between the agent and the teacher.	55
3.3	Idealistic evolution of workload, performance, and autonomy over time for autonomous learning, feedback-based teaching, Wizard-of-Oz (WoZ) and SPARC (arbitrary units).	57
4.1	Setup of the interaction. The application target, a child has been replaced by a robot for repeatability.	65
4.2	Setup used for the user study from the perspective of the human teacher. The child-robot (left) stands across the touchscreen (centre-left) from the wizarded-robot (centre-right). The teacher can oversee the actions of the wizarded-robot through the Graphical User Interface (GUI) and intervene if necessary (right).	68
4.3	Screenshot of the interface used by the participants, the GUI on the left allows to control the robot and a summary of the actions' impact is displayed on the right.	70
4.4	Aggregated comparison of performance and final intervention ratio for both conditions. Dots represent individual datapoint (N=10 per condition) and shaded area the probability distribution most likely to lead to these points.	75
4.5	Evolution of intervention ratio over time for both conditions and both orders. Shaded area represents the 95% CI (N=5 for each plot).	76
4.6	Performance achieved and final intervention ratio separated by order and condition. For each order, the left part presents the metric in the first interaction (with one condition) and the right part the performance in the second interaction (with the other condition).	77
4.7	Questionnaires results on robot making errors, making appropriate decisions and on lightness of workload.	77

5.1	Presentation of different steps in the environment. (a) initial state, (b) step 1: bowl on the table, (c) step 3: both ingredients in the bowl, (d) step 4: ingredients mixed to obtain batter, (e) step 5: batter poured in the tray and (f) step 6 (success): tray with batter put in the oven. (Step 2: one ingredient in the bowl has been omitted for clarity, two different ingredients could be put in the bowl to reach this state)	87
5.2	A representation of the timeline experienced by participants according to the order they were in. The top row corresponds to group 1 and bottom row to group 2.	90
5.3	Comparison of the teaching performance for the six sessions (the left columns present the data of participants in group 1 and the right ones those in group 2). The colours are swapped between session 3 and 4 to represent swapping of conditions. A 6 in teaching performance shows that the participant reached at least one success during the teaching phase. The vertical grey lines represent minimal barplots of the data and the shaded areas the probability distribution most likely to produce these results.	95
5.4	Comparison of the testing performance for the six sessions. A 6 in performance shows that the taught policy led to a success.	96
5.5	Comparison of the teaching time for the six sessions. At 25 minutes, the session stopped regardless of the participant stage in the teaching.	97
5.6	Comparison of the number of inputs provided by the participants for the six sessions.	98
5.7	Comparison of the number of failures for the six sessions.	98
5.8	Average workload for each participants as measured by the NASA-TLX for each conditions in both interaction order.	100
6.1	Setup used in the study: a child interacts with the robot tutor, with a large touchscreen sitting between them displaying the learning activity; a human teacher provides supervision to the robot through a tablet and monitors the robot learning.	113
6.2	Example of the game. Animals have energy in green and have to eat plants or other animals to survive. The child's task is to keep animals alive as long as possible.	114
6.3	Test screen to evaluate children's knowledge, empty starting screen (a) and fully connected and correct test (b).	115
6.4	Simplified schematics of the architecture used to control the robot. In addition, all the topics related to the actions are recorded using rosbag. (Figure adapted from Lemaignan et al. 2018).	117

LIST OF FIGURES

6.5 Example of computation of accumulation. When the effect is true (green), the value increases, when the effect is false (red), the value decreases and the state is set to 0.5 when the effect's value changes.	120
6.6 GUI used by the teacher to control the robot and respond to its suggestions. The game is in the same state as in Figure 6.2, and the robot proposes to move the frog close to the fly (text bubble, arrow, moving the <i>ghost</i> of the frog - a partially transparent image of animal used to determine the position of a move action - and highlight the frog and the fly).127	
6.7 Presentation of different steps in the interaction. It should be noted that steps (b) to (e) correspond to one test and (h) to (k) to one round of the game. As such, a full interaction would see a first pretest (steps b to e), followed the tutorial (steps f and g), two rounds of the game (2 repetitions of steps h to k), a second test, two other rounds of the game and a last posttest before ending the interaction with step (l).	132
6.8 Comparison of distribution of actions executed by the robot in the autonomous and supervised conditions. The left graph is a violin plot of the data, while the right presents the means and the 95% Confidence Intervals.137	
6.9 Children's performance for the three tests: pretest, midtest and posttest for the three conditions.	138
6.10 Children's normalised learning gain after interacting with the robot for the three conditions.	139
6.11 Number of different eating interactions produced by the children (corresponding to the exposure to learning items) for the four rounds of the game for the three conditions.	139
6.12 Points achieved by the children in each round of the game for the three conditions.	140
6.13 Interaction time for the four rounds of the game for the three conditions. The dashed red line represents 2.25 minutes, the time at which unfed animals died without intervention, leading to an end of the game if the child did not feed animals enough.	141
6.14 Summary of the action selection process in the supervised condition: the 'teacher selection' label represents each time the teacher manually selected an action not proposed by the robot.	142
6.15 Number of different actions used by the teacher throughout the interactions with the children.	143

LIST OF FIGURES

Chapter 1

Introduction

Human-Robot Interaction (HRI) explores the challenges of developing robots that interact in human environments, and studies how human beings react to such robots. While not being present in most homes yet, robots are quickly penetrating society and entering people's life. To interact in human environments and take up their role in society, social robots require a way to interpret their sensory inputs and act on the social world; in other words, they need to be sociable (Breazeal, 2004). However, interacting socially with humans is a complex task: the social interaction happens on multiple levels and combines elements from various fields (Fong et al., 2003). Furthermore, by interacting with humans, robots are subjects to social norms and have to take them into account when behaving around people (Bartneck & Forlizzi, 2004).

Providing robots with such a complex policy is a challenge, and no static controller programmed manually would cover all the possible situations they could face. To interact meaningfully with humans, robots need to learn, constantly expanding and refining their policy. As such, it stands to reason to use the experts in humans interactions, which are the humans themselves, to teach robots to interact socially.

By providing control over the robot's actions to a human teacher, a robot could learn efficient policies for social interactions from *in situ* human tutelage and supervision. Interactively learning from humans presents a change of paradigm compared to traditional machine learning, as the human possesses knowledge they can interactively transfer to the robot, allowing for online and adaptive teaching and learning (Fails & Olsen Jr, 2003; Amershi et al., 2014). Finally, by framing this work in the context of HRI we added to the interaction numerous challenges: high-stakes environment (an environment where incorrect actions can have serious consequences), large state and action spaces, relatively low number of datapoints, and real time interaction. Consequently, by exploring how to learn in such a complex environment this research work could have applications beyond HRI, e.g. in general robotics or Machine Learning (ML).

1.1 Scope

As shown in Goodrich & Schultz (2008) only a subset of HRI is designed to be inherently social. Using Goodrich and Schultz's definition of HRI, as the study of robots for use by or with humans, an important part of the field does not consider the social side of the interaction. Robots are often considered as non-social entities having a task to complete or simply tools to be used by humans. However, as demonstrated by Fincannon et al. (2004), even when used as tele-operated machines, robots still provoke social reactions and expectations from humans. Fong et al. (2003) make, among others, the distinction between "evocative robots" and "socially interactive robots". Evocative robots rely on human tendency to anthropomorphise to succeed in their task, while for "socially interactive robots" the social side of the interaction is key. As such, in HRI, the sociality of the interaction might not be intended or even desired but may simply emerge as a by-product of the interaction. As a consequence, in this work we will refer to "social robots" as robots interacting in humans environments and considered as social agents by the humans interacting with them. The intentions of the robot designer or the actual robot's sociality are of minor importance if the humans interacting with the robot project social competencies or expectations on the robot. Regardless of the initial intentions, a robot interacting with humans needs to manage these expectations to ensure the safety and comfort of its human partners. In summary, as soon as a robot interacts with humans, the social side of the interaction needs to be taken into account, i.e. the robot needs a social policy.

However, enabling a robot to interact socially with humans is a complex task. The robot needs to make sense of sensory input, from low level sensory feedback (e.g. joint angles or camera pixels) to high level concepts (e.g. hand-coded events or learned features). And based on the interpretation of these inputs and rules defining a behaviour, the robot needs to select an action or a plan to achieve an assigned goal. As this task covers most of the areas of robotics (Fong et al., 2003), this section will define the scope of the research conducted in this thesis.

1.1.1 Frame

The context of the research conducted here is finding a way for robots to interact efficiently with humans. It has been argued by many researchers (Dautenhahn, 2004; Billard et al., 2008) that robots need to adapt to their users and be able to learn

from them. Social robots cannot simply apply a one-size-fits-all policy programmed in advance by engineers and suited to every possible interaction partner. Robots need the capability to learn interactive behaviours, improving their skills as they inhabit the world and personalising their policy to their users. This thesis aims to tackle this challenge and to provide a framework enabling robots to learn to interact with people.

Throughout this thesis we make the assumption that people knows how robots should interact. A person, generally not the engineers who have developed the robot, possesses some expertise which should be transferred to a robot. This expertise is not related to theoretical knowledge in ML, robotics or other scientific fields but instead is about how the robot should behave. For instance, it could take the form of the steps of a therapeutical intervention a robot should deliver to a child (known to a therapist) or more simply a senior's preferences and desires concerning a robot companion's behaviour. As a consequence, robots' users should be able to teach their robot without requiring technical expertise.

The ultimate goal of this research is to explore how robots could learn to interact socially and efficiently with humans from observing tele-operation of the robot. More specifically, this work addresses the problem of action selection: assuming a perception of the world and a finite set of predefined actions, the robot has to find the best action for each situation. The problems of defining actions (how to execute a high level action), extending the number of actions available to the robot and refining the perception are not covered by this work, but would be interesting future or complementary research. For example, a robot companion at home would learn when to do actions (such as setting the table or cleaning the dishes) but not the motor commands required to complete such actions (we assume the robot has access to an implementation of each available actions).

By making the assumption that the interaction knowledge should be provided by a human, the resulting question of this thesis is: '*How can humans teach robots to interact?*'. As such, this research work aims to provide a convenient and efficient way for domain experts to inculcate robots with their interaction knowledge. To do so, this research explores the requirements on an interaction framework allowing humans to teach robots to interact socially in human environments and studies the impacts such a framework would have on the interaction.

1.1.2 Type of Interaction

In HRI, as in other robot applications, a robot has to solve a task with or for a human, we coin this interaction the *application interaction*. However, by focusing the research on using humans to teach robots, we add a second human-robot interaction to this application interaction: the *teaching interaction* between the robot and its teacher. The end goal of this new teaching interaction is to achieve high standards in the application interaction. This results in two intertwined human-robot interactions: the application interaction, between the application target and the robot, and the teaching interaction, between the teacher and the robot (cf. Figure 1.1). This is equivalent to a general triadic interaction between the human target, the robot and the human teacher.

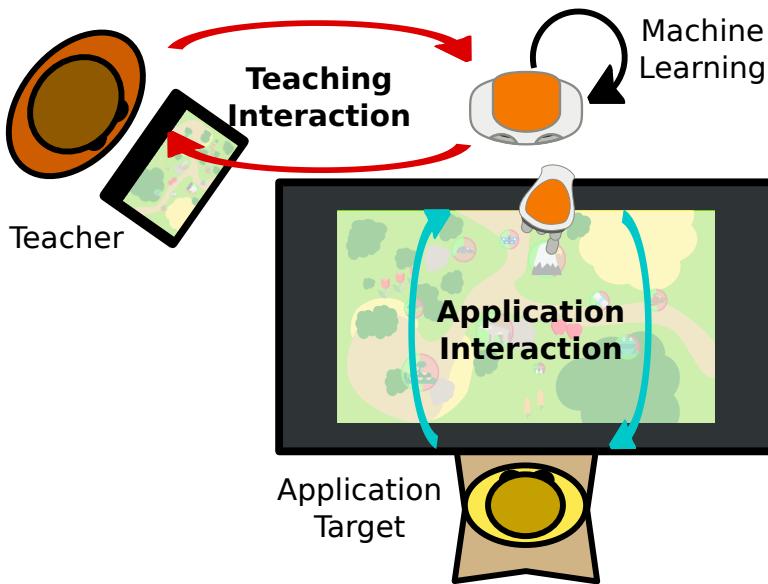


Figure 1.1: The typical interaction used in this research: a robot interacts with a target in an application interaction and learns from a domain expert through a teaching interaction.

The applications presented in this work focus on social HRI as the application domain. However, the methods and algorithms presented in this thesis are applicable to the broader domain in which a robot, or more generally an agent, learns a behaviour, more precisely what action (or label) it should select from a finite set. However, interacting with a human is a high-stakes scenario, where wrong decisions can have serious consequences, where datapoints for learning can be costly to acquire, and with a high diversity between different partners. Other types of interaction could be less constrained, less complex or with lower stakes. Consequently, by addressing the challenging task of teaching a robot to interact socially with humans, the methods and algorithms developed

in this work should also be applicable in other environments. In the more general case, we consider scenarios where a user has a task they would like an artificial agent to complete and they have useful knowledge they want to transfer to this agent. Through the teaching interaction, the human can supervise the agent, demonstrating what it should do and teaching it an efficient policy in the application interaction with the target.

1.2 The Thesis

The main thesis this document proposes is the following:

Through receiving supervision from a human teacher, a robot can progressively and efficiently learn a policy for interacting with people, during the learning the robot will at all times display correct behaviour.

By giving a human control over a robot's behaviour and allowing this human to provide demonstrations, the robot could progressively and efficiently build up a policy by interacting in the real world. As the robot's policy improves, the requirement of demonstrations from the human decreases until the robot's policy is sufficiently appropriate to interact autonomously if desired.

This thesis raises a series of additional research questions; and, in the pursuit of supporting the main thesis, this research work addressed the six research questions presented below.

The first two questions correspond to a theoretical analysis of the field of social HRI and a proposition of an interaction framework allowing humans to teach robots in high stakes environments such as HRI.

RQ1 *What are the requirements of a robot controller for social HRI?*

By interacting with humans, robots enter the social world and have to conform to human expectations while ensuring they can reach their goal. This research question studies the constraints put on the robot's controller by interacting in the human world.

RQ2 *What interaction framework would allow a human to teach a robot while meeting the requirements from RQ1?*

This research question explores the principles that would lead to an efficient teaching interaction between a robot and a human whilst meeting the requirements from RQ1 for the interaction between the robot and the target.

Then, questions 3 and 4 are focused on the teaching interaction and evaluate the specific relation between the teacher and the robot. As such, these questions correspond to situations where the application interaction is fixed and repeatable. These controlled application interactions help us to characterise the teaching interaction, and analyse if it would be suited to teach a robot to interact with humans.

RQ3 *Could a robot decrease its supervisor's workload by proposing actions based on observing the supervisor's earlier decisions?*

Controlling a robot can be a significant burden for the human supervisor. This question explores whether including a learning component in the robot behaviour and proposing actions to the human could relieve or alleviate their workload by taking over some of the mental and physical requirements of the robot control. Additionally, this question explores whether this learning impacts the performance in the task.

RQ4 *When teaching a robot, how does the control over the robot's actions by the teacher impact on the teaching process in terms of performance, effort and teaching time?*

Teaching robots has a unique property compared to teaching humans in the fact that the teacher can have total control over the learner's actions. This question explores whether this control could lead to a faster and more efficient learning while lightening the process for the teacher.

Finally, questions 5 and 6 evaluate the implications of having a human directly teaching a robot to interact with another human in a real situation, more precisely in the context of child tutoring. In this context, teaching the robot to interact with humans creates a triadic interaction where three agents (the teacher, the robot and child) are learning and changing their behaviours over time. These questions are specifically evaluating the Supervised Progressively Autonomous Robot Competencies (SPARC), the interaction framework proposed by this work to address RQ2 (cf. Chapter 3).

RQ5 *What impact does SPARC have on the performance of the child-robot interaction, the comfort for the teacher and the robot's learning?*

Using humans to teach robots to interact with other humans results in two linked interactions. This question studies the impact of the teaching process on these two interactions: are the requirements from RQ1 validated in the application interaction? Is the teaching interaction efficient and convenient for the teacher? Is the robot learning quickly an efficient policy?

RQ6 *After having been taught using SPARC, could a robot behave autonomously in a social context?*

With SPARC, a robot could be deployed after a teaching phase to interact autonomously in the real world. This question analyses the robot's behaviour when interacting autonomously and explores the efficiency of a teacher *in situ* to provide the robot an autonomous behaviour validating principles from RQ1.

1.3 Research Approach

1.3.1 Review of the State of the Art in HRI

The first step toward answering the RQ1 and identifying the requirements interacting with humans puts on robots was to review the different fields of application of social HRI. In each type of interaction, the robot will interact with people with different specificities and needs, and by analysing these fields, we can draw the requirements such interactions put on the robot's controller. For this survey, we limited the applications to cases where humans consider the robot as a social agent, as this is the main feature we use to define social HRI.

1.3.2 New Teaching Framework

To address the requirements on a robot controller emerging by addressing RQ1, we propose the SPARC, a novel teaching framework intended to provide a way for non-experts in robotics to safely teach a robot to interact with humans. SPARC is the cornerstone of this thesis and this research work describes and justifies this approach, and evaluates its impact through three studies of increasing complexity.

1.3.3 Study Design

To address the main thesis of this work and RQ3 to RQ6, we designed and ran three studies, which are described in detail in Chapters 4 to 6. These studies evaluated the impacts of SPARC on different parts of the teaching and application interactions. The first two studies focused on the teaching interaction, i.e. the impact of SPARC on the teachers and the relation between the teacher and the robot. As a consequence, they required repeatable environments to compare humans' efficiency and experiences when teaching a robot. The last study applied SPARC to a real HRI and was run in schools with a single human teaching a robot to interact with children over multiple sessions. Its main goal was to explore whether SPARC can be used to teach a robot to interact with humans. All the studies used SPARC in the type of interaction presented in Figure 1.1, i.e. a teacher wants a robot to complete a task (included but not limited to human interaction) and can control the robot's actions through an interface. As each study focused on different features of SPARC, bespoke interaction environments and test benches have been developed for each study.

The first study (Chapter 4) explored if SPARC could be used to provide learning to a robot in a Wizard-of-Oz (WoZ) interaction and whether this learning would impact the workload and performance of the teacher in the task. This study was designed around a task commonly found in Robot Assisted Therapy (RAT): emotion recognition for children with Autism Spectrum Disorder (ASD) (Dautenhahn & Werry, 2004). In our version of the task, a child would be presented images on a touchscreen and would have to classify them as displaying a positive or negative emotion. On the other side of the screen, a robot can provide some support to help the child to improve in the task. For this study, we decided to replace the child interacting with the robot by a second robot running a model of a child displaying typical features observed in children (such as non-deterministic behaviour, partial observability of state and absence of easily definable optimal policy). This repeatable environment allowed us to evaluate in a controlled study the impact of SPARC on the teacher and compare SPARC to WoZ.

The second study (Chapter 5) compared SPARC to another Interactive Machine Learning (IML) method: Interactive Reinforcement Learning (IRL) in a replication of the environment initially used to test it (Thomaz & Breazeal, 2008). On a computer screen, a virtual robot is in a kitchen and has to learn a sequence of actions to bake a cake.

Additionally, a participant can provide rewards and guidance to help the robot to learn faster. This environment is deterministic with more than 10,000 states, between 3 and 7 actions per state and with an optimal policy reaching a success in 28 steps. The main difference between our method and the one proposed in the original paper resides in the quantity of control provided to the teacher; unlike IRL, SPARC provides teachers with full control over the robot's actions.

Finally, the last study (Chapter 6) deployed SPARC in a real interaction with humans, to teach children about food chains. This study used the paradigm described in Figure 1.1: a child is playing an educational game with a robot on a large touchscreen and the robot is remotely controlled and taught by an adult using a tablet. That way, using SPARC, the robot can learn to support the child in the educative activity and increase their engagement and performance in the task.

1.3.4 Learning Algorithms

To enable robots to learn from humans, they need to be equipped with Machine Learning (ML), in this case an algorithm statistically learning a mapping between some inputs (perceived state) and outputs (possible actions). Two main categories of ML are relevant for this research: Supervised Learning (SL) and Reinforcement Learning (RL).

SL aims to learn a mapping between inputs and outputs to automatically reproduce a desired and known behaviour. It uses a dataset of labelled examples and optimises a mapping to minimise the prediction error of labels (Russell & Norvig, 2016). Typical examples of SL used in this study are Artificial Neural Networks (ANNs) and nearest neighbours methods. ANNs loosely model the way brains and neurons work by having a group of interconnected “neurons” influencing each other. By changing the strength of the connections between the neurons, the network learns to reproduce the desired values on the output nodes according to the inputs. On the other hand, nearest neighbours methods are instance based methods. They compare the distance in a feature space between a point to classify and the different instances stored in a dataset and select the value of a majority of nearest neighbours in that space (Cover & Hart, 1967). With its goal, reproducing a desired behaviour, SL is especially connected with the work carried out in this research. By learning to reproduce the teacher's policy, the robot should reach an efficient behaviour in the target application.

Unlike SL which reproduces a known behaviour, RL aims at providing an agent with the capacity to discover and explore the world it inhabits and learn by interacting in this environment how to behave (Sutton & Barto, 1998). The agent has access to a description of the state and actions it can do, and depending on the action selected, the state will change and the agent will receive a reward. The goal of the agent is to find an optimal policy maximising a notion of cumulated reward. RL is also linked to the work presented in this thesis: by allowing an agent (here a robot) to learn by interacting a world (here human environments), we would reach our goals of efficient human-robot interactions.

Due to the relevance of both fields to this research (enabling robots to learn to interact), algorithms from both categories have been used. The first study presented in Chapter 4 used a feed forward neural network learning to reproduce a teacher policy. The second study in Chapter 5 used RL to combine human guidance with environmental and human rewards to learn an efficient policy. The last study presented in Chapter 6 used an instance based algorithm adapted from Nearest Neighbours to enable online quick and efficient learning from human commands. More details about each algorithms and their related work can be found in the associated chapters.

1.3.5 Terminology

Throughout this thesis, the terms ‘wizard’, ‘supervisor’, ‘user’, ‘expert’ and ‘teacher’ have been used interchangeably to represent the people in control of a robot’s actions and teaching that robot a policy. Similarly, we refer to the robot as ‘robot’, ‘agent’ or ‘learner’ depending on the context. The intended meaning of terms will be clear from the context.

Additionally, the term ‘policy’ is used many times throughout this manuscript. A policy can be understood as a mapping from states to probabilities of selecting each possible action (Sutton & Barto, 1998). A policy can be deterministic (if for any given state, one action has a probability of 1 and the others a probability of 0) or stochastic; in any case, a policy is sufficient to determine an agent’s behaviour.

1.4 Contributions

This research work contributed both scientifically (by creating and evaluating SPARC) and technically (by developing software for multiple projects) to the state of the art in HRI and especially in teaching robots to interact with humans. This section highlights

the original contribution of this thesis and indicates the relevant chapters and published work.

1.4.1 Scientific Contributions

- The first contribution, and the cornerstone of this work is **Supervised Progressively Autonomous Robot Competencies (SPARC)**, a new interaction framework to **teach a robot to interact in complex and high stakes environments**, whereby the robot progressively gains autonomy while being supervised by a human. (Chapter 3; Senft et al. 2015a,b).
- We then **evaluated SPARC in three studies** of increasing complexity, culminating into a final experiment in which we **taught a robot to support child learning** in the wild. (Chapters 4, 5 and 6; Senft et al. 2015b, 2017a, 2018a).
- We showed that by **learning from a human's commands**, a robot could reduce the need for the human to manually select actions it should execute, thus **reducing the workload on this human operator**. (Chapter 4; Senft et al. 2015b)
- We demonstrated the impact and importance of the **teacher's control over the robot's action** when teaching a robot: it **makes the teaching safer, faster, easier and more efficient** (Chapter 5; Senft et al. 2016b, 2017a).
- We designed a lightweight algorithm **adapted from Nearest Neighbours**, but defined on a **sliced version of the state and using rewards** to quickly **learn from demonstrations in multidimensional environments** (Chapter 6; Senft et al. 2017b).
- We used SPARC to **safely teach robots social autonomy** from *in situ* human supervision in a **real human-robot interaction** (Chapter 6; Senft et al. 2018a). This is one of the main contributions as allowing a robot to learn to interact with people *in situ* presents many advantages compared to other types of learning (e.g. **constantly appropriate behaviour in the learning phase, fast learning and transparency**) and can lead to an **autonomous behaviour similar to one resulting from human control**. However, this learning *in situ* is seldomly used in HRI due the challenges it presents.

1.4.2 Technical Contributions

- Software development for three different studies to evaluate SPARC in three different applications, with dedicated interface for the teacher, and combining SPARC with a different algorithm each time.
- Implementation of a new algorithm to learn quickly from human demonstrations.

1.4.3 Other Technical Contributions

- Partial development of a cognitive architecture and two tools for the DREAM project (European FP7 project: 611391) (Appendix A; Esteban et al. 2017).
- Development of an autonomous robot controller to support cardiac rehabilitation in the Human-Robot Interaction Strategies for Rehabilitation based on Socially Assistive Robotics project (Royal Academy of Engineering: IAPP\1516\137) (Lara et al., 2017; Casas et al., 2018).
- Development of a wizard interface for the Freeplay-Sandbox¹ (Lemaignan et al., 2018; Wallbridge et al., 2018).

1.5 Structure

The structure of this thesis is described below with an overview of the content for each chapter. Chapters 3 to 6 start with a summary of the key elements to clarify their contribution.

- This chapter introduced to the general scope and context of this research (robots learning to interact with and from humans), presented the main thesis put forward by this work as well as the technical and scientific contributions and the additional research questions addressed throughout the thesis .
- Chapter 2 provides a description of the different fields of social HRI and draws requirements for controllers of robots interacting with humans. In a second part, it provides an overview of the current robot controllers used in HRI identifying that no current controller fits these requirements. Finally, it introduces IML and proposes to apply it to HRI as a way to validate the previously defined requirements.

¹<https://github.com/freeplay-sandbox>

- Chapter 3 proposes a new interaction framework, SPARC, as a way to apply IML to HRI while validating the three requirements proposed in Chapter 2. This chapter describes the principles behind SPARC and the expectations and limits this method could have.
- Chapter 4 presents a first study evaluating the impact of SPARC on the supervisor's workload and performance compared to WoZ. Results showed that SPARC allowed teachers to achieve a good performance, while reducing their workload.
- Chapter 5 presents a second study comparing SPARC to IRL, another interaction framework from IML. The main difference between the two approaches lies in the amount of control the teacher has: only with SPARC, the teacher has full control over the actions executed by the robot. Results from a 40 participants study supported that this control improves the efficiency of the learning (improving the performance, reducing the time and the number of inputs required to teach as well as decreasing the risks taken during the teaching phase and imposing a lower workload on the teacher).
- Chapter 6 presents the first study where SPARC has been applied to a real world HRI, child tutoring, and to our knowledge one of the first time online learning has been used to teach robots to interact with humans. Results demonstrate that, while not impacting the learning gain of the session, a supervised robot elicited richer child behaviour compared to a passive robot. Furthermore, by learning using SPARC, an autonomous robot reached a policy similar to the teacher's one and achieved similar impacts on the children. These results support SPARC as a teaching method allowing to transfer *in situ* a social and technical policy from a human to a robot in a safe way and leading to an efficient autonomous behaviour. These findings are key for this thesis as they demonstrate the applicability of SPARC to high-stakes, unpredictable, multidimensional, multimodal and complex environments.
- Chapter 7 presents a discussion of the main findings from the previous chapters in relation to the research questions introduced in this chapter. It also discusses the limitations and future directions of research for SPARC as well as its potential impact on HRI and other fields.

- Chapter 8 concludes the thesis by presenting a summary and stating the main contributions.

Chapter 2

Background

This chapter describes social Human-Robot Interaction (HRI) and presents related research in agent and robot control. The first section introduces the different fields of application of social HRI and draws from them requirements for controlling a robot interacting with humans (the robot should: only execute appropriate actions, and have a high level of adaptivity and autonomy). The second section provides the current state of the art in robot control for HRI and analyses it through the requirements presented in the first section. Finally, building on the lack of controller satisfying the requirements from Section 1, the third section presents Interactive Machine Learning (IML), an alternative method holding promise to teach agents to interact, and how it could be applied to HRI.

2.1 Social Human-Robot Interaction

2.1.1 Fields of Application

Reviews of HRI and socially interactive robots have already been presented in Fong et al. (2003), Goodrich & Schultz (2008) and Sheridan (2016). However, these reviews are starting to be dated and/or did not have the same focus than this research. As mentioned in the introduction, this work is focused on social robots, robots perceived as social agents and interacting in a human-centred environment. Depending on the context of interaction, these social robots will have been designed to interact socially with humans, or sometimes, the perception of sociality and agency will simply emerge from the interaction (Fincannon et al., 2004). As such, the following criteria on the interactions were used to select the subfields of HRI relevant to this research:

- Presence of an interaction between a robot and a human (collocated or not): both the human and the robot are influencing each other's behaviour.
- The human partner considers the robot as a social agent, responding socially to the robot and potentially bonding with it.

We reorganised and edited Goodrich and Schultz's categories to fit with the criteria presented above, which resulted in five subfields involving social interactions between robots and humans:

- Socially Assistive Robotics (SAR)
- Education
- Search and Rescue, and Military
- Hospitality, Home and Entertainment
- Collaborative Robots in Industry

We updated each category with recent research and highlighted the social component of the interaction and its impact on the controller used for the robot.

2.1.1.1 Socially Assistive Robotics

Socially Assistive Robotics (SAR) is a term coined by Feil-Seifer & Matarić (2005) and refers to a robot providing assistance to human users through social interaction. This field has been defined by Tapus et al. (2007) as one of the grand challenges of robotics.

One of the main applications of SAR is care for the elderly. In the near future, the ageing of population will have large impacts on the world, and societies will have to find solutions to tackle this challenge. The United Nations' Department of Economic and Social Affairs (2017) reports that the “population aged 60 or over is growing faster than all younger age groups”. This unbalance of growth will decrease the support ratio (number of workers per retiree) forcing societies to find ways to provide care to an increasing number of people using a decreasing staff. Robots represent a unique opportunity to compensate for this lacking workforce potentially allowing elderly people to stay at home rather than joining elderly care centres (Di Nuovo et al., 2014) or simply to support the nursing staff (Wada et al., 2004).

The second main application of SAR is Robot Assisted Therapy (RAT). Robots might be used to provide therapy or to follow and support a patient during their rehabilitation to improve their health, their acceptance in society or recover better from an accident. In therapies, robots were first used as physical platforms to help patient in rehabilitation therapy during the 80s (Harwin et al., 1988). During this period, robots were primarily used as mechatronic tools helping humans to accomplish repetitive task. But in the late

90s, robots started to be used for their social capabilities. For example the AuRoRA Project (Dautenhahn, 1999) started in 1998 to explore the use of robots as therapeutic tools for children with Autism Spectrum Disorder (ASD). Since AuRoRA, many other projects, such as the DREAM project¹, have started all around the world to use robots to help patient with ASD (Diehl et al., 2012; Esteban et al., 2017).

RAT is not limited to ASD only, this use of robots is also explored in hospitals, for example to support children with diabetes (Belpaeme et al., 2012), to support elderly with dementia (Wada et al., 2005) and stroke recovering patients (Matarić et al., 2007) or to monitor and provide encouragement in cardiac rehabilitation therapies (Lara et al., 2017).

These examples demonstrate that already today, robots are interacting with vulnerable populations (children, the elderly or patients in therapy) both in research and as a product². This implies that robots' social behaviours need to be constantly correct to ensure that no harm will be caused to these populations. And due to the shortage of workforce (e.g. nurses in the US; Nevidjon & Erickson 2001), these robots also need to be as autonomous as possible.

2.1.1.2 Education

Robots are being used in education, supporting teachers to provide learning content to children and transforming the teaching process from passive learning to active learning (Linder et al., 2001). In education, social robots can take multiple roles such as peer, tutor or tool (see Mubin et al. 2013 for a review and Belpaeme et al. 2018 for a more recent one). It should be noted that Mubin et al. stress that the intention of the robotic community is not to replace teachers by robots, but provide them with a new form of technological teaching aid. Nevertheless, Verner et al. (2016) presented positive results from an early stage study using a tall humanoid robot to deliver a science lesson to children. The remaining of the section will describe more in details the roles of tutor and peer robots (the 'tool' role has been excluded due its general lack of perceived agency and the lack of social interaction between the robot and the students).

¹<https://www.dream2020.eu>

²currently deployed: <http://zorarobotics.be/index.php/en/> and <http://www.parorobots.com/index.asp>
in development <http://www.embodyied.me/>

Robotics tutors providing tailored teaching content in 1 to 1 interaction. Individualised feedback has been shown to increase the performance of students (Cohen et al., 1982; Bloom, 1984). However, tutoring requires a larger trained staff and as such is more costly than large classes supervised by a single teacher. For this reason, tutoring is seldom used in public education today. Tutoring is however often available through private teacher at additional cost for the parents, potentially increasing social inequalities (Bray, 2009). Robotic tutors could provide this powerful one to one tailored interaction to every student during school time, leading to higher learning gain for the children without dramatically increasing the workload on teachers or the price of education (Kanda et al., 2004; Leyzberg et al., 2012; Kennedy et al., 2016b; Gordon et al., 2016). In addition to classroom uses, robots could also support children at home as they have been shown to elicit advantages over web or paper based instructions (Han et al., 2005).

Peer robots learning alongside the children. Unlike other types of agents in education, peer robots have the opportunity to fit a special role seldom present in education today: the role of care-receiver rather than care-giver (Tanaka & Matsuzoe, 2012). Peer learning has demonstrated benefits both for the helpers and those helped in Human-Human Interaction (HHI) (Topping, 2005). In HRI, a peer robot does not mentor a child to teach them new concepts, but learns alongside or from them, supporting them during the process and encouraging these children to produce behaviours improving their own learning. For example, in the Co-Writer project (Hood et al., 2015), a child has to teach a robot how to write, and as the child demonstrates correct handwriting to the robot, they improve their own skills at drawing letters. Peer robots are able to leverage the concept of learning by teaching (Frager & Stern, 1970) and peer learning (Topping, 2005) in a way hardly matched by adults. The robot can take the role of a less knowledgeable agent with endless patience and encourage the student to perform repetitive tasks such as handwriting. In a similar position, an adult would not be a believable agent requiring learning and a younger student might not have the compliance and the patience to learn from another child.

To provide efficient tutoring or peer support, robots need to be able to personalise their behaviour to the humans they are interacting with in order to maximise the humans' learning gain (Leyzberg et al., 2014). Additionally, as pointed by Kennedy et al. (2015a),

a robot too social could decrease the children's learning compared to a less social robot if its behaviour is not congruent, consequently the robot's social behaviours have to be carefully managed to ensure its effectiveness.

2.1.1.3 Search and Rescue, and Military

Robots are already deployed in the real world, outside of labs and used during search and rescue missions (Casper & Murphy, 2003; Murphy, 2004). For instance, after a natural or artificial catastrophe, robots have been sent to analyse the damaged area and report or rescue the surviving victims of the incident. During these missions, robots have to interact socially with two kinds of human partners: the survivors and the rescue team. In both cases the social component of the interaction is key. The survivor is probably in a shocked state and the searching robot could be the first link they have with the external world after the accident (Murphy et al., 2008). In this case, a social response is expected from the robot and it has to be carefully controlled. On the other side, the rescue team monitoring the robots is under high pressure to act quickly and faces traumatic events too. Even if the robot does not display a social behaviour, rescuers interacting with it might develop some feeling toward the robots they are using during these tense moments (Fincannon et al., 2004).

Similar human behaviours (i.e. emotional bonding with a tele-operated robot) have also been observed in the army (Singer, 2009). Robots have been deployed as tele-operated drones and ground units alongside soldiers to complete scouting task or cleaning minefields. By interacting with a robot for extensive periods, some soldiers developed feelings toward this robot they used in a daily basis: taking pictures with it and introducing it to their friends. These relationships have gone as far as soldiers risking their own life to in order to save the robot used by their squad (Singer, 2009).

These examples in these two fields demonstrate that even in interactions where the robot is not supposed to interact socially with humans, its users might consider it as a social agent. Consequently, the sociability of the robot has to be taken into account when interacting in such stressful environments. Overlooking the importance of social relationships humans will form can lead to dramatic consequences (e.g. soldiers taking risks for the robot). As such, during the interaction, the robot's behaviour needs to always be appropriate not to create misleading expectations and to ensure that the goal of the interaction will be met.

2.1.1.4 Hospitality, Home and Entertainment

Robots are also interacting with humans as receptionists³ and in hotels around the world; for example, the Relay robot (Savioke⁴) delivers amenities directly to the guests' rooms in more than 70 hotels in the US, Europe and Japan⁵. Whilst the social interaction is still minimal today, these robots interact everyday with humans and have been seen evoking social reactions from them⁶.

Since the late 90s, scientists explored how robots could guide visitors in museums and exhibitions (Thrun et al., 1999; Burgard et al., 1999). These researches continue today to explore how to improve the social interaction between tourists and these guide robots (Bennewitz et al., 2005). In a similar context, researchers have designed and tested a robot as a receptionist in a hall at Carnegie Melon University (Gockley et al., 2005). Studies have also explored long term interactions and how humans perceive and interact with robots in shops and shopping malls (Kanda et al., 2009) and how robots should behave with clients (Kanda et al., 2008).

Finally, robots have also entered homes and family circles: from vacuum cleaners to companion passing by pet robots. A notable example is Aibo: twenty years ago, Sony created Aibo, a robotic dog to be used as pet in Japanese families and a new version was released in early 2018⁷. An analysis of online discussions of owners published 6 years after Aibo's first introduction gives insights on the relationship that owners created with their robots (Friedman et al., 2003). For example 42% of the community members assigned intentionality to the robot, such as preferences, emotions or even feelings. Similar behaviours have also been observed when the robot was used not as a pet, but even just as a tool. For instance, Fink et al. (2013) reported that one of their participants was worried that their Roomba (a robotic vacuum cleaner) would feel lonely when they would be away on holiday. More recently, the Pepper robot has been sold as a social robotic companion to families in Japan⁸. However, as of early 2018, no study in English has reported results of the interactions with families or long term use and acceptance.

³<https://www.welbo.eu/about/>

⁴<http://www.savioke.com/>

⁵<https://www.spectrum.ieee.org/view-from-the-valley/robotics/industrial-robots/ces-2018-delivery-robots-are-fulltime-employees-at-a-las-vegas-hotel>

⁶<https://www.fastcodesign.com/3057075/how-savioke-labs-built-a-robot-personality-in-5-days>

⁷<https://aibo.sony.jp/en/>

⁸<https://www.softbankrobotics.com/emea/en/robots/pepper>

By entering our homes or hotel rooms, robots are penetrating some of our most intimate social spaces. These private spheres are ruled by a different set of social norms than public spheres such as streets or shopping mall (Weintraub, 1997) and social faux-pas in these private environment can lead to an important loss of trust. As a consequence, robots' behaviours and policies needs to be especially appropriate and transparent when interacting in such private spaces. Additionally, robots in hospitality and entertainment will face a wide range of users' expectations, and will have to react to different unanticipated behaviours. Consequently, robots needs to be able to adapt their behaviours to these different users. Finally, robots in these fields will also interact with the same people over long periods (Leite et al., 2013). And, to sustain engagement over such time scales, these robots need to change their behaviour over time to overcome possible boredom due to the vanishing of the novelty effect in repeated interactions (Salter et al., 2004).

2.1.1.5 Collaborative Robots in Industry

In industry, robots used to be locked behind cages to prevent humans to interact with them and getting hurt. However, recently social robots, such as Baxter (Guizzo & Ackerman, 2012), have been designed to collaborate with humans; they share the same workspace and interact physically and socially with factory workers. However, to collaborate efficiently with humans, many challenges still need to be addressed. For example, the bidirectional communication between the robot and the humans needs to be as clear as possible. To interact efficiently and safely with humans, robots need to make their intentions clear to humans surrounding them (Dragan et al., 2013) and reciprocally, they also need to interpret humans' social cues to infer their goals and intentions (Scheutz et al., 2007).

Beyond legibility and interpretation, another key challenge in Human-Robot Collaboration (HRC) is task assignment. If a goal has to be achieved by a human-robot team, the repartition of tasks should be carefully managed to optimise the end result in term of task performance, but also to ensure comfort for the human. Explicit and implicit rules and personal preferences describe the expected role and behaviours of each participant and these rules have to be taken into account in HRC. As such, the task repartition system and other planners used in HRC should be aware of these social norms and follow them (Montreuil et al., 2007). For example, recent work done in the 3rd Hand

project⁹ explored how a robot should support a human in a collaborative assembly task by adapting its behaviour to this human's personal preferences and how this adaptation improves the team's efficiency (Munzer et al., 2017).

An last challenge related to the recent advances in Machine Learning (ML), and especially with the omnipresence of neural networks and deep learning, is Explainable Artificial Intelligence (XAI) (Wachter et al., 2017). As agents learning to interact will make mistakes and behave unexpectedly from time to time, they have to be able to provide explanations for these errors in a way understandable by humans. This challenge is especially visible in HRC where both humans and robots aim to collaborate to complete a task together. Hayes & Shah (2017) propose to achieve transparency through policy explanation, by allowing robots to answer questions and explain their behaviour by self observation and logic deduction. This transparency aims at increasing trust between the agents involved in the interaction and improve the team's efficiency.

As demonstrated in Munzer et al. (2017), intelligent systems involved in HRC should adapt their behaviour to their interaction partners, be aware of preferences and rules to follow to ensure that the robot's behaviour is always appropriate, efficient and safe for the humans involved in the interaction. Furthermore, their autonomy needs to encompass behaviour explanation, be able to explain the reasoning steps and justify each of their actions.

2.1.2 Requirements on Robots Interacting with Humans

The review in Section 2.1.1 demonstrated that robots are already interacting with vulnerable populations: young children, the elderly, people requiring healthcare or in a stressful situations (victims of catastrophes or soldiers for example). Additionally, people tend to a create emotional bonding with these robots even if they are not interacting socially with their users. As such, the behaviour of robots interacting with humans need to be carefully controlled to manage humans' expectations and ensure their safety. In other words, robots need to constantly behave in socially acceptable manners, avoiding any confusing, inappropriate or dangerous behaviour. Failure to do so might prevent the interaction from fitting its goal, potentially elicit offence, anger, frustration, distress or boredom, or even lead to physical injuries.

⁹<http://3rdhandrobot.eu/>

These undesired behaviours may come from different origins: lack of sensory capabilities to identify necessary environmental features, lack of knowledge to interpret human behaviours appropriately, failure to convey intentions, impossibility to execute the required action or incorrect policies. Due to the wide range of origins of these potential social faux-pas, this research focuses only on the last point, obtaining an appropriate policy: assuming a set of inputs, finding a way to have the robot select an appropriate action. The other issues are either orthogonal and would lead to failure even with an ‘optimal’ policy as external factors prevent the robot from solving the problem or could be handled by having a better policy (e.g. a policy generalising more efficiently or selecting suboptimal actions when the optimal ones are not available).

Appropriate actions are highly dependent on the interaction context: they could aim to match or reduce users’ expectations of a robot’s behaviour or complete a specific task. Additionally, robots have to follow social norms, and actions correct in a certain context might be inappropriate in another one. Nonetheless, the intention behind being *appropriate* here is that actions executed should be guaranteed not to present risks for the humans involved in the interaction (for example, preventing physical harm or mental distress), while helping the robot to move toward its goal and achieving its objectives.

Additionally, interactions with humans ‘in the wild’ (Belpaeme et al., 2012) do not happen in well defined environments or rigid laboratory setups. In the real world, robots have to interact in diverse environments, with a large number of different people, for extended periods of time or with initial incomplete or incorrect knowledge. As such, the policy also needs to be adaptable to the context and users as well as evolve over time. In summary, robots also need to be adaptive, to react to changing environments, cover a larger field of application and improve their policy over time.

Lastly, in many cases today, interactive robots are not autonomous but partially controlled by a human operator (Riek, 2012; Baxter et al., 2016). We argue that to have a real and useful HRI, the robot needs to be as autonomous as possible. Robots are expected to be used in areas where the workforce is already in shortage (e.g. healthcare) and requiring humans to control these robots decreases widely their applicability. As a community, HRI should strive toward more autonomy for robots interacting with humans.

Based on these considerations, we define three properties to evaluate how suited robot controllers are to interact with humans. Each axis is associated to a principle the robot has to follow to sustain meaningful social interactions:

1. Appropriateness of actions - The robot should only execute appropriate actions.
2. Adaptivity - The robot should adapt to its environment and its users, and be able to learn.
3. Autonomy - The robot should require as few human inputs as possible.

We will use these axes to analyse current robotic controller types in Section 2.2.

As HRI is a large field, other research axes are equally important, such as the complexity or the depth of the interaction, the constraints put on the environment, the ability of the robot to set its own goals, the dependence and knowledge of social rules or the range of application of a robot to cite only a few. However, we did not address these axes in the current work as they are more influenced by the goal and the context of the specific human-robot interaction taking place than the policy itself. Additionally, an appropriate, adaptive, and autonomous robot should be able to safely and autonomously learn to interact in deeper and more complex interactions and learn to extend its abilities beyond the ones it initially started with, thus increasing its range of applications.

2.1.2.1 Appropriateness of Actions

As argued previously, much of social human-robot interaction takes place in stressful or sensitive environments, where humans have particular expectations about a robot's behaviour. Additionally, even in less critical situations, human-human interactions are subject to a large set of social norms and conventions resulting from precise expectations of the interacting partners (Sherif, 1936). And some of these expectations are also transferred to interactions with robots (Bartneck & Forlizzi, 2004).

We define appropriate actions as actions correct in terms of both the task and social context, i.e. actions taking into account the social side of the interaction, and producing a correct robot behaviour at the right time. This behaviour needs to be safe for surrounding humans and help the robot to reach its goal. Failing to produce these appropriate actions, for example by not matching the users' expectations, may have a negative impact on the interaction, potentially compromising future interactions if the human feel disrespected,

confused or annoyed. Furthermore, failing to behave appropriately can even harm the people interacting with the robot: not reminding an elderly to take their medication, not taking into account the state of mind of survivors after a disaster or behaving inconsistently with children with ASD might lead to dramatic consequences. Robots require a way to ensure that all the actions they execute do not present risks to the humans involved in the interaction while moving the robot closer to its goal.

For the review in Section 2.2, the appropriateness of actions axis is a continuous spectrum characterising how much the system controlling the robot ensures that the robot constantly acts in a safe and useful way for the users. For example, a robot selecting its actions randomly would have a low appropriateness as no mechanism prevents the execution of unexpected or undesired actions. On the other hand, a robot continuously selecting the same action as a human expert would have a high appropriateness as domain experts would know which action is the correct one in this application domain.

2.1.2.2 Adaptivity

Humans are complex, nondeterministic and unpredictable agents, as such an optimal robot behaviour is not likely to be known in advance or even programmable by hand (Dautenhahn, 2004; Argall et al., 2009). Specifically, end users will express behaviours not anticipated by the designers, the interaction environment is often not perfectly definable and the desired behaviour might also need to be customisable by or for the end user or evolve over time. While many studies in HRI use robots following a static script, to interact meaningfully outside of lab settings or scientific studies, robots need this flexibility to extend their range of application and improve their interactions with users. In other words, robots interacting with people need to be able to adapt their policy to the environment and improve their behaviour over time. We use the term *adaptivity* to represent this ability to express a behaviour suited to different conditions and refine it over time.

We propose three components for this adaptivity. The basic one is the adaptivity to the environment, i.e. the generalisation of the behaviour (reacting accordingly to unseen inputs). The second one is personalisation and adaptation: being able to adapt a behaviour to the current user or context. Finally, the last component is learning, the ability to enrich and refine the policy over time.

Generalisation: Robots are interacting in human centred environments which are complex and highly stochastic. These environments are often under specified and robot designers cannot explicitly anticipate every single possible human reactions or occurring events. Furthermore, the state representations often use large vectors with multiple possibilities for each values. As such, predefining a specific robot reaction for each state possibility or each possible human behaviour is not feasible. Consequently, robots should have a policy able to generalise to unseen and unexpected situations and react appropriately to different environments.

Personalisation and adaptation: As robots are interacting with humans, they will encounter different types of environment, contexts of interactions and persons with different roles. For example a robot used as an assistant for elderly people will have to interact in the home of the owner, but also follow them in the street or in a supermarket. In these different interaction contexts, distinct behaviours will be expected from the robot. Similarly, different human beings might have different roles and the robot needs to adapt its policy to the type of person it is interacting with. For instance, in education, an autonomous robot would have to interact both with the students and the teacher, and its behaviour needs to take into account the role of the people it is interacting with. Additionally, the robot needs to personalise its behaviour to the person it interacts with: in entertainment or search and rescue, none of the users are known beforehand but providing a personalised behaviour adapted to the context may significantly impact the outcomes of the interaction. In summary, robots need to be able to adapt their policy to the environment and context they interact in and personalise their behaviour to the different users and their status.

Learning: Lastly, robots should also be able to learn, updating their policy over time to extend their range of abilities and improve their efficiency. While the personalisation does include an evolution of the behaviour over time, this could be achieved with a fixed policy (for example by using rules as proposed in Leyzberg et al. 2014). However, we believe that robots' adaptivity should go beyond a fixed set of rules influencing behaviours, robots need to be able to learn and update their policies over time.

Additionally, providing a robot with the capacity to learn enables it to be used by non-experts in robotics, granting them a way to design their own human-robot interactions, and making use of their expertise and knowledge to have their robot interacting as they

desire. This is crucial as many application of social robotics, such as RAT, happen in environments where non-technical people possess the domain expertise required to ensure that the robot is efficient. And, as stated by Amershi et al. (2014), learning reduces the requirements of expensive and time consuming rounds-trips between domain-experts and engineers and additionally decreases the risks of confusion between these different communities.

In summary, for this review, adaptivity is a continuous scale ranging from no adaptivity at all (the robot has a linear script that it follows in all the interactions), to high adaptivity (the robot can generalise to unforeseen situations, dynamically change its policy according to the context of interaction and its partners, learn new actions and tasks, and improve its policy over time).

2.1.2.3 Autonomy

Today, many experiments in HRI are conducted using a robot tele-operated by a human (Riek, 2012; Baxter et al., 2016), and whilst having a human controlling the robot presents many advantages (e.g. the human provides the knowledge and the adaptivity required to interact efficiently and has sensing and reasoning capabilities not yet available for robots), multiple reasons push us away from this type of interaction (Thill et al., 2012). First, relying solely on tele-operation is not suited for deploying robots in the real world. Human control does not scale to interact for long periods of time or in many places. Robots are also expected to interact in fields already lacking workforce (e.g. healthcare), so if robots need to be continuously controlled, the advantage of automation is highly reduced. Second, for research, using humans to control robots reduces the transferability of knowledge gain to the real world. The human-robot interaction tends to become “a human-human interaction mediated by a ‘mechanical puppet’ ” (Baxter et al., 2016), which decreases the relevance of the robot as an agent and as such reduces the applicability of studies’ results to future interactions with fully autonomous robots. Finally, humans are biased in their decisions; and when controlling robots, such human biases will impact the robots’ behaviour, decreasing the replicability of results. While these biases can be partially addressed by specifying clearly the wizard’s behaviour, they nonetheless remain an issue Riek (2012). For these reasons, among many, we argue that a robot used in social HRI should be as autonomous as possible. A limited human supervision could support the robot and be used to improve its behaviour, but

the robot should not rely on humans for its action selection during the main parts of the interaction.

To analyse the different robot controller's autonomy, we take inspiration from Beer et al. (2014). Beer et al. define three components of autonomy: sensory perception, analysis and action selection. As such, autonomy is organised following a spectrum of different levels from no autonomy at all: a human is totally controlling the robot (doing sensory perception, analysis and action selection) to a full autonomy: the robot senses and acts on its environment without relying on human inputs. Levels exist between these extremes where a human and a robot share perception, decision and/or action: for example the robot can request information from a supervisor or the supervisor can override the action or goal being executed (Sheridan & Verplank, 1978).

2.1.2.4 Interdependence of Factors and Trade-Offs

These three axes used for the review: appropriateness of action, adaptivity and autonomy are not independent. Especially, as a robot able to learn might be also able to improve its appropriateness of action and its autonomy as it refines its policy. This impact of adaptivity on the two other axes is fundamental to increase the robots' range of application, performance and usability. However, while a learning robot could eventually reach an optimal, perfect and autonomous policy, the behaviour expressed by the robot in early stages of the learning, while the policy is not appropriate yet, is critical. Even during this learning phase, the robot's behaviour needs to be safe and useful for humans interacting with it. As such, adaptivity is a key element for a robot controller to improve and reach a correct and autonomous policy, but a mechanism must ensure that at every step of the interaction the robot's behaviour is appropriate regardless of the learning progress.

2.2 Current Robot Controllers in HRI

The previous section presented three axes to evaluate a robot controller: the appropriateness of actions, the range of adaptivity and the level of autonomy. Based on these three axes, this section presents and analyses high level categories of robot control currently used in HRI to define a robot behaviour. For each category, we will present the corresponding approach, indicate representative works done and qualitatively rate it on the three axes.

2.2.1 Scripted Behaviour

One of the simplest ways to have a robot interacting with a human is to have an explicit and fixed behaviour. In this case, the robot is fully autonomous and follows a script for action selection. Success in using this approach is dependent on having a well defined and predictable environment to have the interaction running smoothly. If the interaction modalities (possible range of behaviours and goals) are limited enough, a constant policy can be sufficient to handle all (sensible) human actions.

This approach is followed in a large number of research in HRI: as many studies are human-centred, the focus is not in the complexity of the robot's behaviour but on how different humans would interact with and react to a robot displaying a behaviour with defined and controlled specificities. This also allows researchers to compare conditions with controlled differences and analyse the impact of small variations of behaviour. Whilst this has advantages when exploring people's reactions to robots, this method can hardly be used to deploy robots to interact with humans on a daily basis. Real world applications take place in undefined and open environments where potential human behaviours are almost infinite. Additionally, a fixed robot behaviour might also create boredom in users once the novelty effect vanishes (Salter et al., 2004).

By essence, this type of controller has no adaptivity as the robot is following a preprogrammed script, but is fully autonomous as no external human is required to control the robot; and when the application domain is highly specified, the behaviour can be mostly appropriate.

2.2.2 Adaptive Preprogrammed Behaviour

To go beyond a simple script, robots can also be programmed to react in predefined ways to expected human actions. By adaptive preprogrammed behaviour, we denote a behaviour programmed before the interaction, but explicitly (or implicitly) including ways to adjust the policy in reaction to anticipated human behaviours. This preprogrammed adaptation takes two forms: either having a fixed number of variables impacted by the actions of the partner and guiding the policy (for instance using homeostasis), or explicitly planning for specific behaviours to be produced if predefined conditions are met (for example using a finite state machine).

Homeostasis, the tendency to keep multiple elements at equilibrium, is constantly used by living systems to survive and is also a good example of the first case of preprogrammed adaptation used in social HRI. For instance, Breazeal (1998) used a set of drives (social, stimulation, security and fatigue) which are represented by a variable each and have to be kept within a predefined range to represent a ‘healthy’ situation. If these variables reach values outside the desired homeostatic range, the robot is either over or under-stimulated, this will affect the robot’s emotional status and it will display an emotion accordingly. This behaviour have been shown to be enough to maintain human interacting with the robot for relatively long periods of time spanning more than ten minutes. Homeostasis approaches have also been extended to robotic pets (Arkin et al., 2003) or RAT (Cao et al., 2017).

On the other hand, a case of planned adaptation is clearly presented in Leyzberg et al. (2014). Participants have to play a cognitive game, and a robot delivers predefined advices on strategies depending on the performance and the current lack of knowledge of the participant. With these anticipated human behaviours, the robot can provide personalised support as long as the participants behave within expectations.

Similarly to other behaviour-based methods used in robotic control (such as the subsumption architecture; Brooks 1986), due to the indirect description of behaviours, homeostasis-based methods are more robust in unconstrained environments than a purely scripted controller, while remaining totally autonomous. However, as the policy is fixed, the robot does not learn, which limits its adaptability to unanticipated situations. Furthermore, with this indirect description of actions, there is no guarantee against the robot acting in inconsistent way in specific cases which limits the appropriateness of actions. Similarly, planned adaptation provides adaptivity to the environment but only in highly limited cases expected by the designers. This limits the adaptability of such an approach as the robot does not learn; and, as the robot may face situations not expected by the designers, the maximum appropriateness of actions is also limited.

In summary, both predefined adaptation and homeostasis-based methods score highly in autonomy and can have a moderate to high level of appropriateness, but the adaptivity is low as they can only adapt to the environment within predefined, anticipated and limited boundaries and the robot does not learn.

2.2.3 Wizard of Oz

Wizard-of-Oz (WoZ) is a specific case of tele-operation where the robot is not autonomous but at least partially controlled by an external human operator to create the illusion of autonomy in an interaction with a user. It outsources the difficulty of action selection and/or sensory interpretation to a human operator. This technique has emerged from the Human-Computer Interaction (HCI) field (Kelley, 1983) and is today common practice in HRI (Riek, 2012). Similar to scripted behaviours, Wizard-of-Oz (WoZ) is mostly used in human-centred studies to explore how humans react to robot and not as a realistic way to control robots in the wild. A second use of WoZ is to safely gather data to develop a robot controller from human demonstrations (cf. Section 2.2.4).

Even in WoZ, part of the robot's behaviour is autonomous, and combining this robot autonomy and human control can be done in multiple ways. Baxter et al. (2016) define two levels of WoZ related to the levels of autonomy presented by Beer et al. (2014) and that correspond to the level of human involvement in the action selection process. Cognitive WoZ aims to provide a robot with human-like cognition or deliberative capabilities; while in perceptual WoZ, the human only replaces a sensory system and feeds information to the robot controller. Typically, perceptual WoZ replaces challenging features of the controller required for a study, but not relevant to the research question. One of such typical challenge is Natural Language Processing (NLP). Despite all the progress made in speech recognition, NLP is still a challenge in HRI, especially when interacting with children (Kennedy et al., 2017). And as some studies require a limited speech recognition element to test an hypothesis, using a human for that part of the interaction allows to run the study without having to solve complex technical challenges (for instance, see Cakmak et al. 2010).

This level of human control impacts the autonomy of a system: a robot relying on human only to do perception has a higher autonomy than a robot fully controlled by an operator. Controllers can also combine human control and predefined autonomous behaviour in mixed systems. For example, Shiomi et al. (2008) propose a semi-autonomous informative robot being mainly autonomous, but with the ability to make explicit request to a human supervisor in predefined cases where the sensory input is not clear enough to make a decision.

With WoZ, the adaptivity and the appropriateness of actions are provided almost exclusively by the human, so these characteristics are dependent of the human expertise but are generally high. However, due to the reliance on human supervision to control the robot, the autonomy is low. For semi-autonomous robots, the picture is more complex: as explained by Beer et al. (2014), the initiative, the human's role and the quantity of information and control shared influence the level of autonomy. For example, in Shiomi et al. (2008) the robot explicitly makes requests to the human, but the human cannot take the initiative to step in the interaction, thus limiting the adaptivity (especially as the robot policy is fixed). And as the human only has limited control over the robot's behaviour, no mechanism prevents the robot to make undesired decision. Overall, this would lead to a higher autonomy, but a lower appropriateness of actions and adaptivity compared to classical WoZ.

2.2.4 Learning from Demonstration

As stated by numerous researchers, explicitly defining a complex behaviour and manually implementing it on a robot can take a prohibitive amount of time or even could not be possible for complex behaviours (Argall et al., 2009; Billard et al., 2008; Dautenhahn, 2004). This statement applies equally well to manipulation tasks and social interaction. In both cases, humans have some knowledge or expertise that should be transferred to the robot. However in social robotics, experts in the application domain often do not have the technical knowledge to implement efficient behaviours on a robot, which results in numerous design iterations between the users and engineers to reach a consensus.

The field of Learning from Demonstration (LfD) aims to tackle these two challenges: implementing behaviours too complex to be specified in term of code and empowering end-users with limited technical knowledge to transfer a policy to a robot. The learning process starts with a human demonstrating a correct behaviour (Argall et al., 2009), and then offline batch learning is applied to obtain a policy for the robot. Later, if required, reinforcement learning can complement the demonstrations to reach a successful policy (Billard et al., 2008). In LfD, the human-robot interaction is key, however in most of the cases this interaction is only in the learning interaction, the application interaction does not involve humans, but often manipulation or locomotion tasks such as grasping and moving an object, using a racket to hit a ball or throwing tasks (Billard et al., 2008).

However, two approaches have applied LfD to teach robots a social policy to interact with humans. The first one aims at learning directly from human-human interactions and replicate these human behaviour on a robot. For example, Liu et al. (2014) present a data driven approach taking demonstrations from human-human interactions to gather relevant features defining human social behaviours. Liu et al. recorded motion and speech from about 180 interactions in a simulated shopping scenario and then clustered these behaviours into high-level actions and implemented them on the robot. During the interaction, the robot uses a variable-length Markov model predictor to estimate the probability of a human executing each actions and finally winner-take-all is applied to select the most probable action. According to the authors, the final robot's behaviour was life like but not perfect. However, authors affirm that if this approach were scaled using a larger dataset gathered from more human-human interactions in the real world, the performance should improve and become closer to natural human behaviours.

The other approach is to collect data through a WoZ setup and aims to learn to replicate the wizard's policy to reach an autonomous social behaviour. Knox et al. (2014) coined this approach "Learning from the Wizard (LfW)". The method starts with a purely WoZ control study to gather data, and then, a policy is derived by applying machine learning on the collected data. However, this original paper did not present a description of which algorithm could be used or how, and did not evaluate the approach, instead it only offered a reflection on the application of this idea. An implementation and evaluation is briefly discussed by the authors in Knox et al. (2016), but the lack of implementation details and results reduces the usability of the paper.

LfW is widely used in HCI (especially dialogue management; Rieser & Lemon 2008) and has also been implemented by other groups of researchers in robotics. For example, Sequeira et al. (2016) extended and tested this idea to a create a fully autonomous robot tutor. Their method is composed of a series of steps:

1. Collect observations of a human teacher performing the task.
2. Define the different actions used by the teacher and the features used for the action selection.
3. Implement these actions and features on a robotic system.

4. Set up a restricted-perception WoZ experiment where an operator uses only the identified features to select actions for the robot.
 5. Combine machine learning applied on the data and hand-coded rules to create an autonomous robot controller.
 6. Deploy the autonomous robot.
- (7. If required, add offline refinement steps to fine tune the robot's behaviour.)

Both Knox et al. (2014) and Sequeira et al. (2016) stress the importance of using similar features for the Wizard of Oz control than the ones available to the robot during the autonomous part. Although this decreases the performance in the first interaction, it allows more accurate learning overall due to the similarity of inputs for the robot and the human controlling it.

Clark-Turner & Begum (2018) aimed to bypass these limitations by using a deep Q-network (Mnih et al., 2015) to learn an Applied Behaviour Analysis policy for RAT. They recorded videos, microphone inputs and actions selected in a WoZ interaction with neurotypical participants to train the network with the raw inputs and the actions selected to obtain a controller able to deliver the therapeutic intervention. However, in their study, the autonomous robot required additional limited human input to inform the algorithm of the state of the therapy and only reached a behavioural intervention with an accuracy inferior to 70%. This means that even with additional human input the robot would provide inconsistent feedback at some points in the interaction. However, this study only used a limited amount of data, and using more training examples should lead to better results.

LfD methods are based on real interactions either between humans, or between humans and robots controlled by humans; and, with enough demonstrations the robot should be able to replicate a human policy, thus select appropriate actions. However, efficiency is limited by the type of inputs recorded, the capabilities of the learning algorithm and the quality of the demonstrations which limit the appropriateness of the policy (as seen in Clark-Turner & Begum 2018). Furthermore, after the learning phase, the robot's behaviour is mostly static, without any additional learning provided. As such, while possessing a good generalisation capability, LfD generally stops learning once the robot is deployed. Sequeira et al. (2016) propose to add offline learning steps, but online

learning would allow for smoother transitions and improvements of the behaviours. Finally, all these methods require the presence of humans in a first phase but the robots are fully autonomous later in the interaction, so the autonomy is low in the first phase and then high during the main part of the interaction.

2.2.5 Planning

An alternative way to interact in complex environments is to use planning (Asada et al., 1986). The robot has access to a set of actions with preconditions and postconditions and a defined goal it needs to reach. To achieve this goal state, it follows the three planning steps: *sense*, *plan* and *act*. The first step, *sense*, consist on acquiring information about the current state of the environment. Then, based on the set of actions available and the goal, a *plan* is created. This plan is a trajectory in the world, a succession of action and states which, according to the defined pre and postconditions, should lead to the goal. Finally, the last step is to *act*, to execute the plan. The plan can be reevaluated at each time step or only if an encountered state differs from the expected one, in that case the robot updates its plan according to the new state of the environment and continues trying.

The efficiency of planning relies on having a precise and accurate set of pre and postconditions for each action. And as humans are complex and unpredictable, it is a serious challenge, if not impossible, to model them precisely. As such, planning have seen limited use for open social interactions with humans. However, due to the nature of planning, reaching a specific goal in a known environment, it has been applied successfully to Human-Robot Collaboration (HRC). Additionally, by limiting the interaction to a joint task, HRC also simplifies the modelling of the human: the interaction being more constrained and task-oriented, the human should limit its behaviour to a number of expected task-related actions. The Human Aware Task Planner (Alili et al., 2009) is an example of planning used to assign task between a robot and human in a HRC scenario. One specificity of this planner is the ability to take into account predefined social rules (such as reducing human idle time) when creating a plan to allocate tasks to the human-robot team. Including these social norms in the plan construction is expected to improve the user experience and maximise human compliance to the plan, which should lead to higher performance in the task.

A precise and correct model would ensure that the robot would autonomously select the appropriate action whilst an incorrect one would lead to non-appropriate behaviours. Similarly, the adaptivity depends on the model the robot has access to and whether it can update it in real time. However, in many cases when interacting with humans, the model is static, only covering the tasks the robot has to complete the different contexts and states it is expected to face and as such presents limited generalisation capabilities to unanticipated situations or non task-related human actions.

Planning has also been extended with learning, which then allows for more adaptive and personalised policies. This type of learning planner has been mostly used in manipulation and navigation to obtain better trajectories (Jain et al., 2013; Beetz et al., 2004) but not exclusively. In HRI, Munzer et al. (2017) presented a planner adapting its decisions to human preferences in a HRC scenario. With this approach, the robot estimates the risk of each actions and depending of the risk value will execute them, propose them (and waits for approval before executing an action), or wait for a human decision. Between repetitions of the task, the robot will update its planner to fit more precisely to the human preferences and improve its policy for the next iteration of the task. Munzer et al. adopted principles from LfD to planning to improve quickly and efficiently the performance of the robot. However, while planning is well suited for strictly defined and mostly deterministic tasks, many social human-robot interactions cannot be totally specified symbolically with clear actions and outcomes and as such the application of planning to social HRI has been limited. Nevertheless, it does provide robots with an autonomous, partially adaptive and appropriate policy.

2.2.6 Summary

Table 2.1 presents a summary of the different approaches currently used in social HRI with their advantages and drawbacks for application in HRI and their evaluation on each of the three axes. The two most promising types of control are LfD and planning, however, both of them have their drawbacks: LfD is applied offline to create a monolithic controller with limited adaptivity after being deployed, and planning's reliance on a model of the world limits its application to open-ended social HRI in the wild.

Similarly to humans, an ideal robot controller would learn how to interact by interacting, by receiving feedback from the environment but also by being taught by humans. Robots have access to characteristics unique to artificial agents, and roboticists should use

Controller	Advantage	Drawbacks	Application in HRI	Appropriateness	Adaptivity	Autonomy
Fixed preprogrammed behaviour	Quick and easy to create Clear specified and repeatable behaviour	Limited to highly constrained interactions	Human-centred studies in highly constrained env.	Low	Null	Maximal
Adaptive preprogrammed behaviour	Relatively simple to program More robust and efficient than scripted behaviour	Only provide adaptability in limited anticipated context	Human-centred studies in constrained environments	Medium	Low	Maximal
Wizard of Oz	Use human expertise to select the best action	Require constant high workload from human Not scalable	Human-centred studies Highly critical HRI	Maximal	Maximal	Null/Low
Learning from Demonstration	Transfer knowledge from human to robot in the real application environment	Lack of learning once deployed	HRI case by case	High	Medium	High
Planning	Complex behaviours and adaptable to variations in the environment	Human too complex to be clearly modelled Limited application to social interaction	Complex defined environments HRC	Medium	High	Maximal

them when designing robots that learn: endless patience, no risk of becoming tired and potential full control by another agent to learn faster from humans. However, while learning is important, an ideal robot behaviour should also guarantee its actions are constantly appropriate. One way could be to use a human to control the robot in early stages of the learning, when the policy is not correct yet. This would ensure an initial appropriate policy in early stages. And in later stages, this robot could combine autonomous learning, and being taught by other agents. This learning from supervised interaction would be the approach with the most potential as this type of learning could validate the three requirements: appropriateness of actions, adaptivity and autonomy. By essence, this continuous online learning aims at providing open-ended adaptivity to the robot. Including a human with control over the robot's actions can also ensure that actions are appropriate. And finally, as the robot learns, accumulates datapoints and demonstrations from the teacher, it improves its policy, reducing the reliance and workload on the human to reach high levels of autonomy while conserving the constant appropriateness of actions and the adaptivity.

This type of interaction is similar to Interactive Machine Learning (IML): learning from the interaction and using a human teacher to speed up the learning. As shown in the next section, researchers have explored how to teach agents interactively non-social policy (Scheutz et al., 2017; Cakmak et al., 2010); but as of early 2018, no controller exists in HRI applying IML to the challenge of learning social interaction with humans in the real world.

2.3 Interactive Machine Learning

In Section 2.1.2.2, we stated that to be adaptive enough, a robot should be able to learn. Consequently, robot controllers used in HRI should make use of Machine Learning (ML). ML corresponds to a field of Artificial Intelligence (AI) aiming at providing artificial agents with learning capabilities to improve their behaviour and reach high level performances in a large range of tasks. ML has two main trends referring to the synchronisation between the learning and the use of algorithm: offline and online learning.

In robotics, offline learning is a technique allowing the robot to change its policy over time by updating it outside of the interaction (such as Learning from the Wizard in Section 2.2.4). Between or before the interactions, a learning algorithm is used on a dataset previously accumulated to create a new policy.

On the other hand, online learning (such as Reinforcement Learning (RL); Sutton & Barto 1998) learns during the interaction. Rather than single monolithic definitions or updates of the behaviour, this constant refinement of the agent's policy benefits from a high number of updates, allowing the robot to learn even during the first interaction with the environment and never stop improving its behaviour.

Interactive Machine Learning (IML), as coined by Fails & Olsen Jr (2003), is a type of online learning with two specificities:

- Use of an end-user expert in the learning process.
- Learn by multitude of consecutive small updates of behaviour.

These two characteristics differ greatly from classical offline learning, such as deep learning (LeCun et al., 2015) which uses costly monolithic learning steps without human influence to define a static behaviour. On the other hand, IML is an iterative process where the behaviour is improved at each small step, and where the end-user can provide feedback on the learner's performance during all these iterations. IML aims to learn faster, by continuously using the human expert to correct the errors made by the algorithm as they appear, to provide additional useful information to the learner and to improve the knowledge gained at each learning step.

Amershi et al. (2014) presents an introduction to IML by reviewing the work done and presenting classical approaches and challenges faced when using humans to support machine learning.

2.3.1 Goal

The main goal behind IML is to leverage the human knowledge during the learning process to speed it up, to extend the use of classifiers from static algorithms trained only once to evolving agents learning from humans and refining their policies over time. As explained in Fails & Olsen Jr (2003), classifiers would gain from using human knowledge to iterate quickly to reach a good solution, and agents learning from the interaction would gain from using additional feedback from humans (Thomaz & Breazeal, 2008; Knox & Stone, 2009). IML aims to combine the advantages from both Supervised Learning (SL) and online learning and applies this new type of learner to classification or interaction tasks.

Furthermore, by allowing a human user to see the output of an algorithms and direct the learning process, IML has the potential to be faster and tailored to this human's desires. By using human expert knowledge and intuition, the system can achieve a good performance faster (Thomaz & Breazeal, 2008). Additionally, a key advantage of IML is also being able to empower end-users of robotic or learning systems. These users are often non-technical, but possess valuable knowledge about what the robot should do. IML provides an opportunity to allow these users to design the behaviour of their robot, to teach it to behave the way they desire.

These human inputs take three forms: labels for specific datapoints (cf. active learning; Section 2.3.2), feedback over actions (similarly to reward in RL; cf. Sections 2.3.3 and 2.3.4) or demonstrations to reproduce (cf. LfD - 2.3.5).

2.3.2 Active Learning

Active learning is a form of teaching used in education aiming to increase students' achievement by giving them a more active role in the learning process (Johnson et al., 1991). This approach has been transferred to machine learning, especially to classifiers, by allowing the learner to ask questions and query labels from an oracle for specific datapoints with high uncertainty (Settles, 2009). The typical application case is when unlabelled data are plentiful, but labels can be limited in numbers or costly to obtain. As such, a trade-off arises between the performance of the classifier and the quantity of queries made by the algorithm. Often, the oracle would be a human annotator with the ability to provide a correct label to any datapoint, but their use should be minimised for reasons of cost, time or annoyance.

Using an oracle to provide the label of specific points with high uncertainty should highlight missing features in the current classifier resulting in improvements both in term of accuracy and learning speed. However, this specific relation between the learner and the human teacher raises new questions such as:

- Which points should be selected for the query?
- How often should the oracle be queried?
- Who controls the interaction? (i.e. who has the initiative to trigger a query - the agent or the oracle?)

Researchers have explored optimal strategies for dealing with this relation between the learner and the oracle. This research has been especially active in HRI with robots directly asking questions to human participants and exploring how the robot's queries could inform the teacher about the knowledge of the learner (Chao et al., 2010). In a follow up study, Cakmak et al. (2010) showed that most users preferred the robot to be proactive and involved in the learning process. On the other hand, they also wanted to be in control of the interaction, deciding when the robot could ask questions even if it leads to a higher workload for them. However, authors proposed that when teaching a complex task requiring a high workload on the teacher, the robot would probably be expected or should be encouraged to take a more proactive stance requesting samples to take over some workload from the teacher.

Active learning, being able to select a specific sample for labelling, can dramatically improve the performance of the learning algorithm (Settles, 2009). However, when interacting in the world, the learner is not in control of which sample can be submitted to an oracle to obtain a label. Datapoints are provided by the interaction and are influenced by the learner's actions and the environment's reactions. For agents learning during the interaction, the active learning approach working for classifiers is not applicable, so other methods have been applied such as RL, learning from human feedback or LfD.

2.3.3 Reinforcement Learning

The main framework to learn from interaction is Reinforcement Learning (RL). RL aims to solve the problem of finding the best policy (i.e. a policy maximising a notion of cumulated reward) by observing the environment reaction to the agent's action.

2.3.3.1 Concept

Young infants and adults learn by interacting with their environment, by producing actions and receiving a direct sensory motor feedback from their environment. By learning the impact of their actions, humans can learn how to achieve their goals. Similarly, the field of RL aims to empower agents by making them learn by interacting, using results from trials and errors and potentially delayed rewards to reach an optimal policy (Sutton & Barto, 1998).

Most of the RL agents interact in a discretised version of the time, considering life as a sequence of states and actions. The simplest version of RL takes place in a

finite Markov Decision Process (MDP), a discrete environment defined by the tuple $\langle (S, A, P_a(s, s'), R_a(s, s'), \gamma) \rangle$ (Howard, 1960), with:

- S : a finite set of states defining the agent and environment states.
- A : a finite set of actions available to the agent.
- $P_a(s, s')$: the probability of transition from state s to s' following action a .
- $R_a(s, s')$: the immediate reward following transition s to s' due to action a .
- γ : a discount factor applied to future rewards.

The goal of the RL agent is to find the optimal policy π_* (assigning an action from A to each state in S) maximising the discounted sum of future rewards. The agent is not aware of all the parameters governing the environment, but only observes the transitions between states and the rewards provided by the environment and has to update its policy to maximise this cumulated reward. Different algorithms exist to reach this policy, but a challenge faced by most of them is to balance the exploration and the exploitation (Sutton & Barto, 1998).

Exploration consists on trying out new actions to learn more about the environment and potentially gain knowledge to improve the policy; whilst *exploitation* is the execution of the current best policy to maximise the current gain of rewards. RL algorithms have to balance these two objectives to reach an optimal policy. One way to deal with this trade-off is to start with high probability of exploration, to rapidly collect knowledge on the environment and then decrease this probability to settle on an efficient behaviour.

The more complex the environment is, the longer the agent has to explore before converging to a good policy. Thus, using RL, it is not uncommon to reach numbers such as millions of iterations before reaching an appropriate policy (Sutton & Barto, 1998). Additionally, when the agent is exploring, its behaviour might seem erratic as the agent tries actions often randomly to observe how the environment is reacting.

2.3.3.2 Application to HRI

This approach presents many features relevant to HRI: it provides both the autonomy and the adaptivity required for having meaningful interactions with humans. However, as explained in the previous section, traditional RL has two main issues: requirement of exploration to gather knowledge about the environment and large number of iterations

before reaching an efficient policy. Generally, RL copes with these issues by having the agent interacting in a simulated world. This allows the agent to explore safely in an environment where its actions have limited impact on the real world (only time and energy) and where the speed of the interaction can be highly increased to gather the required datapoints in a reasonable amount of time. For example to solve heads-up limit poker (Bowling et al., 2015), an agent played two months while considering more than 24 trillions hands every second¹⁰. However, no simulator of human beings exists today which would be accurate enough to learn a policy applicable in the real world. Learning to interact with humans by interacting with them would have to take place in the physical world, with real humans, and this implies that these issues of time and random behaviours would have direct impacts.

To gather information about the environment, the agent needs to explore, trying out random actions to learn how the environment responds to them and if the agent should repeat them later. However, when interacting with humans, executing random actions can have dramatic effects on the users, presenting risks of physical harm as robots are often stiff and strong or cause distress. This reliance on random exploration presents a clear violation of the first principle to interact with humans presented earlier ('Only execute appropriate actions').

Even if random behaviours were acceptable, humans are complex creatures, not fully predictable, with personal preferences and desires. And as such, learning to interact with them from scratch would require a large number of datapoints and as interactions with humans are slow (not many actions are executed per minute) the time required to reach an acceptable policy would be prohibitive.

Despite similar real-world constraints, RL has been used in robotics (Kober et al., 2013), but mostly applied to manipulation, locomotion or navigation tasks. For the reasons stated above, as of early 2018, RL has never been used to fully autonomously learn rich social behaviours for HRI.

2.3.3.3 Opportunities

Despite the limitations presented in the previous section, changes can be made to RL to increase its applicability to HRI. For example, combining RL and IML can ensure that the behaviour is appropriate to interactions with humans even in the learning phase.

¹⁰4000 CPU considering 6 billions hands per second: <http://poker.srv.ualberta.ca/about>

García & Fernández (2015) insist on *safe* RL, ways to ensure that even in the early stages of the interaction, when the agent is still learning about the world, its policy still achieves a minimal level of acceptability. The authors present two ways to achieve this safety: either by using a mechanism to prevent the execution of non-safe actions or by providing the agent with enough initial knowledge to ensure that it is staying in a safe interaction zone. These two methods are not limited to RL but are also applicable to other machine learning techniques to make them safer (for instance LfD; Billard et al. 2008).

The first method (preventing the agent to execute undesired actions) can be implemented by explicitly having ‘forbidden’ actions in predefined states or by having a list of safe actions (Alshiekh et al., 2017). Using this method, the anticipated cases of errors can be prevented. However it seems unlikely that every case could be specified in advance, so such a method might not be sufficient for applying RL to HRI.

The second method (providing enough initial knowledge) can be achieved by carefully engineering the features used by the algorithm or starting from a initial policy to build upon. For example, Abbeel & Ng (2004) proposed to use human demonstrations in a fashion similar to LfD but to learn a reward function and an initial working policy. This method, Inverse Reinforcement Learning, has been applied successfully to teach a flying behaviour to a robotic helicopter. Once the initial policy and a reward function were learned from demonstrations, RL was applied around the provided policy to explore and optimise the policy. That way, only small variations of the policy happened and only around the demonstrated one. These small variations ensured that policies leading to incorrect behaviours were negatively reinforced and avoided before creating issues (such as crashing in the case of the robotic helicopter).

Whilst being promising and having been applied for agents interacting in human environments (such as for personalised advertisement; Theocharous et al. 2015) these approaches have not been used to learn social behaviours or to have robot interacting with humans.

2.3.4 Human as a Source of Feedback on Actions

When an agent is learning in a RL fashion and improves its behaviour by receiving rewards from the environment, an intuitive way to steer the agent’s behaviour in the desired direction faster is to use additional human rewards. This approach is an

adaptation of ‘shaping’: tuning a animal’s behaviour by providing rewards (Bouton, 2007). In ML, using rewards from a human to bias and improve the learning presents multiple advantages which will be presented throughout this section. One notable advantage is the simplicity of the interface and its generalisability to any type of problem. As the teacher only needs a way to provide a scalar or a binary evaluation of an action to steer the learning, only a simple one-way interface is required. However, this simplicity of interaction is associated with a limited efficiency and a complexity of interpretation: the issues of how to interpret human rewards and how to combine them with environmental ones if existent are an active research field today (Knox & Stone, 2010).

When used on their own, human rewards enable an agent to learn a policy even in the absence of any environmental rewards, which is specially interesting robotics as a clear reward function applicable to HRI or robotics in general can be complex to define. Early work in that field came from Isbell et al. (2006) who designed an agent to interact with a community in the LambdaMOO text based environment. Cobot, the agent, had a statistical graph of users and their relations and executed actions in the environment. Users of LambdaMOO could either reinforce positively or negatively Cobot’s action by providing rewards. While the interaction between the agent and the users was limited, Isbell et al. presented the first agent to learn social interactions with humans in a complex and social online environment.

While the goal of Cobot was to create an entity interacting with humans, Knox & Stone (2009) explored how humans could actively teach an agent a policy in the absence of environmental rewards using TAMER (Training an Agent Manually via Evaluative Reinforcement). With this approach, the agent uses a supervised learner to model the human reward function and then takes the action that would receive the highest reward from the model.

However, unlike environmental rewards, human rewards are subjective evaluations of an agent’s behaviour. As such by knowing humans tendencies and intentions when providing rewards, an agent is able to obtain more information from these human rewards than by treating them the same way as environmental ones. Many researchers explored how to obtain more information from human reward. For instance, Advice (Griffith et al., 2013) models how trustworthy the teacher is and as such how much importance the learner should give to their rewards. For example, rewards from inconsistent teachers

will be reduced as the agent knows the source is not reliable. Alternatively, Loftin et al. (2016) explored how to infer the strategy used by the teacher in the reward delivery. Similar behaviours from different teachers might have different meaning: not rewarding an action might reflect an implicit acknowledgement of the correctness of an action or the active refusal to provide a positive reward (indicating the incorrectness of an action). By modelling this intention, the real meaning of rewards can be inferred and used to further improve the learning. Another relevant feature explored by this community is the dependence in time of the human reward policy. While reward functions are generally constant in time with RL, with humans they might vary according to the current performance of the agent. For example, a suboptimal policy could receive positive feedback early on, when it compares positively to the average behaviour; while receiving negative feedback later on, when the average agent's performance is better. MacGlashan et al. (2017) proposed COACH (Convergent Actor-Critic by Humans) to model how humans adapt their rewarding scheme in function of the agent's performance and deal with this non-stationary reward function. Similarly to other factors biasing human rewarding strategies, this dependence of the reward function to the current agent's policy should be taken into account to maximise the knowledge gained from human rewards.

Even when environmental rewards are present, human rewards still have opportunities to improve the learning: they can enrich a sparse reward function, guide the robot faster to an optimal policy or correct incomplete or incorrect environmental rewards. Knox & Stone (2010) explore nine different ways to combine these two types of rewards and each methods' impacts on the learning. From this analysis, they explain how to select an approach according to the specificities of the environment and the reward function.

Teachers can also use rewards to communicate other information to the learner. For example, Thomaz & Breazeal (2008) aimed to explore how humans would use feedback to teach a robot how to solve a task in a virtual environment. They used Interactive Reinforcement Learning (IRL) as a way to directly combine environmental rewards and human ones. However, during early studies, Thomaz and Breazeal discovered that participants tried to use rewards to convey intentions, informing the robot which part of the environment it should interact with. The next study involved two communication channels, a reward one to provide feedback on the actions and a guidance channel to provide information about the action the robot should execute. This guidance has been

actively decided to be ambiguous; participants could not explicitly control the robot, but just bias the exploration, and adding this second channel improved the performance of participants. This study presented a first attempt to combine environmental rewards, human ones and human guidance to teach an agent a policy and demonstrated the importance of giving additional ways for the teacher to impact the robot's behaviour.

While not being applied to robotics but mostly to learning non-social interactions, these implementations of IML provide important research describing how robots could be taught to interact with humans. These human rewards are especially interesting when the environmental reward function is sparsely defined or non-existent, providing a way to teach robots in any environments.

In summary, human teachers desire to go beyond simply evaluating what the agent is doing, they adopt teaching strategies and want to provide advices or commands about how the agent should behave (Amershi et al., 2014).

2.3.5 Interactive Learning from Demonstration

As presented in Argall et al. (2009) and Billard et al. (2008), LfD is majoritarily used in an offline learning fashion to learn a defined task without extending the policy once the task is considered mastered. However, tasks such as social interaction are complex even for humans and probably will never be fully mastered for robots. As such (and as argued before), robots would highly profit from learning throughout all their life, not only once before being deployed, but learning new tasks and improving their skills as often as required (Dautenhahn, 2004).

With Interactive Learning from Demonstration (ILfD), an agent is deployed to interact, but still receives demonstrations if required. ILfD is related to Mixed Initiative or Shared Control (Adams et al., 2004; Kucukyilmaz & Demiris, 2018) where an agent and a human share control on the agent's actions. The robot acts mostly autonomously, but in some cases (at the initiative of the human or the robot), the human takes over the robot control and make a demonstration that will be used by the robot to refine its policy for the future.

One approach highly relevant to this work and giving teachers a total initiative on the interaction is Dogged Learning (DL) (Grollman & Jenkins, 2007). With DL, an agent is autonomously interacting and a teacher has the power to override the agent behaviour at any time by selecting desired actions or outputs. Facing a potential difference between

the algorithm's outputs and the teacher's ones, the robot executes the commands with highest confidence (often the human's one) and the learning component aims at reproducing the executed output. If the teacher does not provide any commands, the ones from the algorithm are used. DL does not provide the robot with the opportunity to request a demonstration, but instead, the robot can communicate its uncertainty to the teacher, indirectly asking for demonstrations.

Alternatively, Chernova & Veloso (2009) propose a method with a more complex interaction between the learner and the teacher. The Confidence Based Algorithm (CBA) is composed of two components: the Confident Execution (CE) and the Corrective Demonstration (CD). The CE enables the agent to act autonomously when its confidence in its policy is high and on the other hand to actively request a demonstration when the confidence is low. The CD allows the teacher to provide a corrective demonstration when the agent executes an incorrect action, which provide more information to the agent than a classic negative reward. These two components aim to leverage the complementary capabilities of the learner and the teacher. CBA has demonstrated efficient teaching in diverse scenario such as simple driving simulators or other classification tasks. But the effectiveness of this approach is bounded by the capacity of the learner to estimate this confidence to be able to request demonstrations and prevent incorrect behaviour to be executed. Another limit of such an approach is the impossibility for the teacher to correct undesired actions before they negatively impact world.

Both methods rely on the teacher being able to anticipate the robot's behaviour to provide demonstrations before an incorrect action is executed or before it impacts the agent and its environment. As such, the appropriateness of the robot controller is not at maximum as the teacher cannot ensure that no incorrect action will be executed during the learning, only that the robot would learn faster from its errors.

2.3.6 Importance of Control

Results from active learning, research using human to provide feedback and LfD have shown that human teachers take an active stance during the training of an agent and want multiple ways to influence the learner's behaviour (Amershi et al., 2014). Humans are not oracles, enjoying providing labels and evaluating an agent's actions, they desire to be in control of the learning and provide richer information to the agent. Kaochar et al. (2011) have shown that when given choice between different teaching methods,

humans will never choose to limit themselves to use only feedback, they want to teach using more modalities; for example, by using demonstrations to define an initial policy and then refine it with feedback.

In addition to improving the teacher's experience in the teaching process, providing the humans with more control improves the learning (Thomaz & Breazeal, 2008; Chernova & Veloso, 2009). By allowing the teacher to demonstrate a policy online, bias the action selection and preempt or correct undesired actions, the learner interacts mostly in useful states of the environment and with a correct policy. This lead to faster learning and would improve the robot's performance highly in early stages of the learning. Another fundamental feature added by this human control over the robot's actions is safety. If a domain expert can prevent a robot interacting with humans to make errors and can ensure that all its actions are efficient, it would increase greatly the quality of the interaction for the humans involved. This will further improve the applicability and use of the robot and would satisfy the two first principles: appropriateness of actions and adaptivity of the robot.

However providing the teacher with this control presents challenges for designing the interaction between the robot and its teacher. Unlike a simple scalar reward, being able to control the robot requires the teacher to be able to give commands or advice to the robot and to receive additional information about the learner beyond its observable behaviour. This enriched two-way communication might be complex to design, especially when the action space is bigger than a few actions or the learning mechanism not transparent. In addition to the communication interface, the time scales of the interaction are also key: to give the opportunity to the teacher to preempt undesired actions, the learner needs to communicate its intentions in a timely manner to the teacher which complexifies the relation between the learner and the teacher.

2.4 Summary

This chapter presented first an overview of the fields of HRI where robots interact socially with humans. From these cases of application, we defined three principles a robot controller should follow. To interact efficiently with humans, the robot should:

1. Only execute appropriate actions.
2. Have a high level of adaptivity and learn.

3. Have a high level of autonomy.

Secondly, a review of current controllers for robots in HRI reported that no approach applied today in the field validates these principles. The review was extended to more general methods in ML with potential to satisfy these principles. IML shows promises for enabling a robot to learn online how to interact with humans, especially when the teacher is given control over the robot's behaviour and can demonstrate a correct policy. However while humans have been used to teach robots behaviours or concepts, teaching them to interact with people in an interactive, online fashion has not been demonstrated in the field so far.

Chapter 3

Supervised Progressive Autonomous Robot Competencies SPARC

Key points:

- Proposition of a novel interaction framework, SPARC, to teach robots a policy while interacting.
- A human teacher is in control of the robot's actions whilst the robot learns from this supervision:
 - The teacher can select actions to be executed by the robot.
 - The robot learns from these demonstrations.
 - The robot proposes actions about to be executed to the teacher.
 - The teacher provides feedback on propositions (i.e. intentions) rather than executed actions, and can preempt actions.
- The robot's behaviour (under supervision) can be assumed to be optimal.
- The workload on the teacher decreases over time as the robot learns.

Parts of the work presented in this chapter have been published in Senft et al. (2015b) and Senft et al. (2017a). The final publications are available from Springer and Elsevier:

- SPARC: Supervised Progressively Autonomous Robot Competencies. In International Conference on Social Robotics¹.
- Supervised Autonomy for Online Learning in Human-Robot Interaction. In Pattern Recognition Letters².

¹http://dx.doi.org/10.1007/978-3-319-25554-5_60

²<https://doi.org/10.1016/j.patrec.2017.03.015>

3.1 Motivation

As presented in Chapter 2, robots would profit from being able to learn from human teachers how to interact with other humans. We propose to use the general Interactive Machine Learning (IML) methodology to achieve this transfer of social and task knowledge from the human domain-expert to the robot. This would result in a faster and safer learning than slow iterative updates of behaviours by engineering the policy, learning from large quantities of data or learning by trials and errors as with Reinforcement Learning (RL).

However, as stated in this Section 2.3, IML has never been applied to teach robots to interact with humans. No current system provides the teacher with enough control over the robot's actions to validate the first principle presented in Section 2.1.2 ('Only execute appropriate actions'). Techniques relying solely on feedback from the teacher cannot prevent the robot to execute an incorrect action, but only reduce the chances of future errors by rewarding negatively incorrect actions after their execution (Senft et al., 2017a). And, with current techniques based on Learning from Demonstration (LfD) the teacher relinquishes its control over the robot when not demonstrating, only reacting in hindsight after the learner makes mistakes and its erroneous actions have impacted the real world (Chernova & Veloso, 2009).

The problem tackled in this research is to provide a robot with an appropriate policy, adaptive to different contexts and partners' behaviours and requiring a low workload on the teacher. As such, in Senft et al. (2015b), we introduced Supervised Progressively Autonomous Robot Competencies (SPARC) framework of interaction. SPARC aims to allow end-users to safely and easily teach a robot a policy applicable to social Human-Robot Interaction (HRI).

3.2 Context

Similarly to other applications of IML, SPARC requires inputs from a teacher to learn a policy to interact with the world. As framed in the introduction, in this framework, the robot interacts with two entities: the target and the teacher (as shown in Figure 3.1). This results into two intertwined interactions: the application interaction (task the robot learns to achieve) and the teaching interaction (relation with the teacher). In the generic case, the overall interaction is a triadic interaction (Teacher - Robot - Human target

or Teacher - Robot - Environment); for instance, a teacher could teach a tutor robot to support child learning in an education task (as implemented in Chapter 6). But in specific cases, the application target can be the teacher himself, such as a robot at home learning from its user how to support them better.

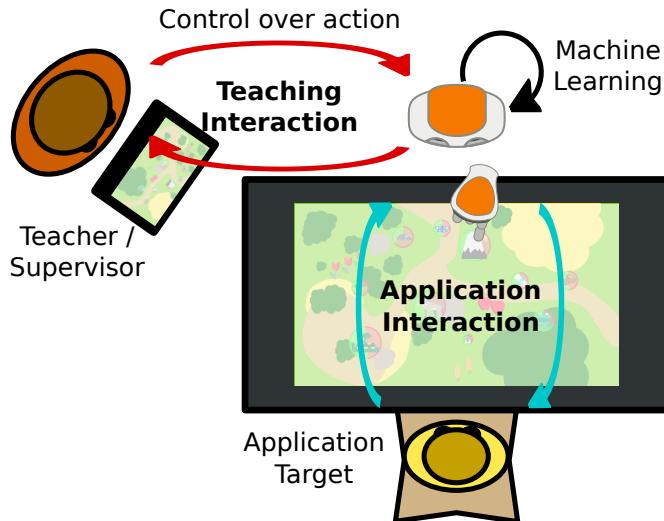


Figure 3.1: Frame of the interaction: a robot interacts with a target, suggests actions and receive commands and feedback from a teacher. Using machine learning, the robot improves its suggestions over time, to reach an appropriate policy.

3.3 Principles

SPARC defines an interaction between a robot (physical or virtual agent) and a teacher following these principles:

- The robot has access to a representation of the state of the world and a set of actions.
- The teacher can select actions for the robot to execute.
- The learner can propose actions to the teacher before executing them (informing them about its intentions).
- The teacher can enforce or cancel actions proposed by the robot and actions non evaluated are implicitly validated and executed after a short delay.
- The robot uses Machine Learning (ML) to improve its policy using the teacher's commands and feedback on propositions.

This type of interaction between the robot and the teacher is similar to the level 6 on the Sheridan scale of autonomy: "computer selects action, informs human in plenty of

time to stop it" (Sheridan & Verplank, 1978); with the addition that the human has also the opportunity to select actions for the agent to execute. In this thesis, we will refer to this interaction as 'Supervised Autonomy': the robot interacts autonomously under the supervision of a human who can ensure that the robot's behaviour is constantly appropriate. This automatic execution without human intervention is an important design decision as it could allow the robot to move towards autonomy and reduce the teacher's workload once the robot starts to learn. Depending on the situation, this time before execution could vary and potentially be set to infinity to force a human to confirm each action in highly sensitive environments.

This way of keeping a human in the learning loop, with the opportunity to override the agent actions, and the robot learning from these demonstrations is similar to Dogged Learning (Grollman & Jenkins, 2007) (cf. Section 2.3.5). However, with SPARC, the robot informs its teacher about its intentions, allowing them to prevent undesired actions from being executed. This results in a mixed control system where the teacher can select actions and have the robot execute them while the robot only proposes actions to the teacher. In response to a suggestion, the teacher has the choice between preempting the action or let it be executed. A learning algorithm on the robot's side uses the feedback and commands from the teacher to improve the correctness of the suggested actions until reaching an efficient policy. This learning mechanism coupled with the auto-execution of actions aims to decrease the requirement of interventions from the teacher over time, thus reducing the workload on the teacher as the robot learns. Additionally, keeping the human in the loop also gives them the opportunity to provide additional information to the algorithm speeding up the learning and to create a mental model of the robot. Algorithm 1 and the flowchart in Figure 3.2 represent in a logic and a graphical way this interaction between the robot and the teacher.

Additionally, The main difference between SPARC and CBA (Chernova & Veloso, 2009) is that with SPARC, the robot communicates its intentions and the teacher has total control over the robot's action. With CBA and other classical IML, the teacher have to wait for an action to impact the world before correcting it or assigning it a negative feedback (Thomaz & Breazeal, 2008; Knox & Stone, 2009). However, using SPARC, the teacher is informed beforehand of the robot future actions and can preempt them before they impact the world. The agent learns to avoid actions with expected negative impacts without having to face the results of their execution. This implies that the

Algorithm 1: Pseudo code of the mixed control between the robot and the teacher with SPARC.

```

inputs :  $s$ : current state
          A: set of actions
           $t$ : time before auto-execution
outputs :  $a$ : evaluated action
            $r$ : reward associated to  $a$ 
           $\pi$ : robot's policy
if Teacher selects action  $a$  then
    Robot executes  $a$ 
     $r > 0$ 
     $\pi$  is updated with  $s-a-r$ 
else
    if Learning Algorithm selects  $a$  then
        Robot proposes  $a$  to teacher
        if Teacher cancels  $a$  before  $t$  then
            Robot does not execute  $a$ 
             $r < 0$ 
        else
            Robot executes  $a$ 
             $r > 0$ 
         $\pi$  is updated with  $s-a-r$ 

```

behaviour executed by the robot can be assumed to be optimal (or at least as good as the teacher's), making the interaction safer and potentially simplifying the learning mechanism.

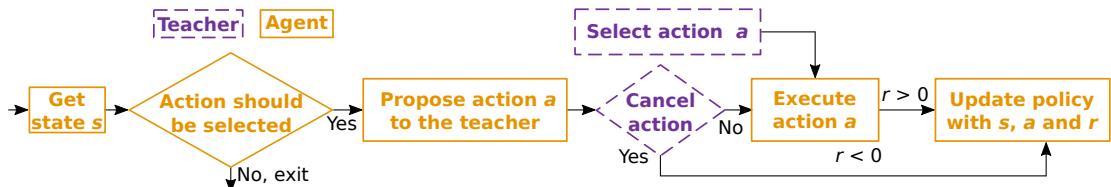


Figure 3.2: Flowchart of the action selection loop between the agent and the teacher.

This approach is comparable to predictive text as seen on phones nowadays. The user can select the words proposed by the algorithms (or implicitly accept them by pressing the space bar) or write their own. The algorithm learns the user's preferences and habits and aims to suggest words more and more appropriate for the user. However, predictive text aims mostly to correct users' errors and interact in a complex static environment. On the other hand, SPARC aims to replicate a teacher's policy in continuous time, in a dynamic and interactive environment evolving both dependently and independently of the agent's actions.

Alternatively, SPARC can be seen as a way to provide proactivity to an agent. By observing interactions on longer time scales, such as an assistant robot at home,

SPARC allows the robot to propose to help its user when the current state is similar to previous observation. This would compare to a passive case where each action executed by the robot has to be requested by the user or an autonomous robot interacting in the house without any transparency. By proposing actions to the teacher, the robot takes the initiative to support humans, possibly reminding its user something they forgot, while not imposing its presence. In human environments, executing actions (such as starting to play music, changing the lighting condition or cleaning the kitchen) without informing the surrounding humans could be perceived as rude or annoying if the timing is not right. On the other hand, this proactivity also needs to be kept in control as a robot proposing to help too often might be equally annoying.

3.4 Goal

SPARC aims to provide an interaction framework to teach robots a policy possessing the following characteristics:

- Be usable by people without expertise in computer science.
- Allow fast policy learning from *in situ* guidance.
- Require little or no human input for the robot to act in the world.
- Constantly ensure an appropriate robot behaviour.

Figure 3.3 presents an idealistic comparison of the expected workload, performance and autonomy of four methods: autonomous learning (e.g. RL; Sutton & Barto 1998), feedback based teaching (e.g. TAMER; Knox & Stone 2009), Wizard-of-Oz (WoZ) (Riek, 2012) and SPARC. Unlike other learning methods, by following the principles presented in Section 3.3, SPARC is expected to maintain a constant high performance even during early stages of learning. In later stages of the learning, the agent keeps improving its policy, making its suggestions more accurate and allowing the auto-execution of actions to reduce the workload on the teacher.

Once the behaviour is deemed appropriate enough by the teacher, the agent is ready to be deployed to interact autonomously in the real world, if this outcome is desired. Alternatively, in contexts where a human expert still cannot be removed from the control loop, such as Robot Assisted Therapy (RAT), a supervisor could stay in control of the robot's actions by using the Supervised Autonomy. Similarly to the teaching phase,

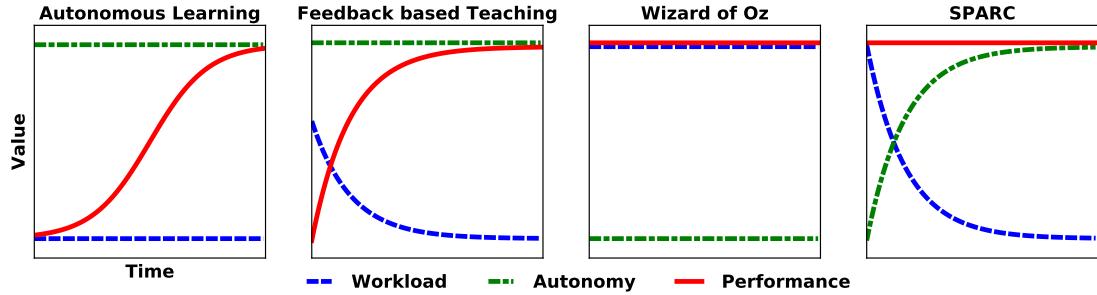


Figure 3.3: Idealistic evolution of workload, performance, and autonomy over time for autonomous learning, feedback-based teaching, WoZ and SPARC (arbitrary units).

with this Supervised Autonomy, the robot would inform the supervisor of its actions and the human would only have to intervene in case of incorrect propositions. While still requiring attention from the supervisor, this reduction of human actions to control the robots would reduce the workload on the supervisor. Furthermore, as the supervisor would accumulate a better knowledge of the agent's behaviour, they could be especially careful in cases where the agent would be prone to making errors. And as the control of the agent would require less effort from the supervisor, they could focus more closely on the application interaction rather than the teaching one.

3.5 Implications

3.5.1 Relation with Time

Similarly to other IML approaches, the requirement of a human in the action selection loop limits the time scale of interaction and this effect is an important consideration when applying SPARC to an interaction. With this type of IML, three time scales are coexisting: the robot's, the teacher's and the interaction's. The robot has an internal clock running probably multiple times per second, sensing the world and deciding if an action should be proposed. The human teacher has to be able to cancel proposed actions before their executions, so they need to be provided with a 'correction window' spanning more than one second to react to propositions. And finally, for some applications, actions are appropriate only during a short time, and if this amount of time (the validity window) is shorter than the correction window, action executed automatically would not be appropriate anymore, reducing the validity of SPARC.

Depending of the application, actions might have to be executed in a time critical environment (such as driving) or less critical ones (such as an assistant robot at home). While being easily applicable to slow interactions, SPARC could still be applied to

these time critical ones. For example, if applied to partially autonomous driving, the interface could display to the driver the planned trajectory with augmented reality and let the driver correct this trajectory if not appropriate. However, more critical elements, such as emergency breaking, would probably have to be done with a much shorter correction window so the car can break in time (and the way information is delivered to the driver should be adapted too). It should be noted that these two applications (HRI and autonomous driving) present different constraints and challenges. Driving is a fast-paced and dangerous application, but with partially predictable outcomes of actions and where the main challenge is interpreting the environment; while social HRI tends to be less critical in terms of risks but also less deterministic, and with optimal policies not evident.

Similarly, there is a trade-off between the granularity of the action and the control provided to the teacher. To provide the teacher with enough time to react to incorrect propositions, only high level actions can go through the approval process. Low level actions require finer control and are more sensitive to time, thus they might be obsolete or inefficient if executed after the time delay required to allow the teacher to review them.

Additionally, the presence of this correction window reduces the rate of actions selection to 0.5 Hz or below, which might reduce the applicability of SPARC compared to other IML methods. However, this limitation of application is the price to pay to ensure the appropriateness of actions, and this effect can be mitigated by using higher level actions or by focusing on applications less critical in term of time.

Finally, as the actions are high level ones, their rate of selection will be much lower than the rate of the robot's action selection loop. At a human level, actions will be executed at a rate of few actions per minutes, while the robot's processing runs at multiple hertz. This indicates that unlike classical RL methods, in most of the steps, the robot should not select any action. When interacting with humans, the learning algorithm and the state representation needs to take into account these differences of time scales to ensure that the robot's behaviour is coherent and useful.

3.5.2 Difference with LfD

SPARC is an Interactive Learning from Demonstration (ILfD) method (cf Section 2.3.5) and as it uses human demonstrations of policies to learn, it presents many similarities with non-interactive LfD techniques (cf. Section 2.2.4). However, most of the applications

of LfD (Argall et al., 2009; Billard et al., 2008) are focused on learning a manipulation skill in a mostly deterministic environment. LfD has seldom been used to teach a policy to interact with humans (Liu et al., 2014; Sequeira et al., 2016; Munzer et al., 2017) and never in an online fashion. Munzer et al. proposed an interactive planner that would learn offline the current user's preferences and desires, but two key differences exist between this approach and SPARC. The first one is the application domain. The learning planner is well suited to clearly defined environments (e.g. Human-Robot Collaboration (HRC)) where a similar task with clear steps has to be done multiple times. As such the learning can happen offline between the repetitions. SPARC is defined to be applicable to non-linear underspecified environments, with less constrained tasks, where the learning should happen online. Secondly, using a threshold, Munzer et al. define actions with low risk which are executed (and can be cancelled during the execution) and actions with higher risk which have to be validated first by the human. SPARC does not make this distinction, but proposes both types of actions to the teacher and will start executing them if no feedback is received. This removes the need for the human to explicitly approve correct proposed actions while ensuring that the human can cancel incorrect actions before their execution. This principle aims at reducing the number of required interventions from the human to teach and interact with the robot compared to methods such as the one presented by Munzer et al.

3.5.3 Interaction with Learning Algorithms

The principles of SPARC define it at the crossroads between Supervised Learning (SL) and RL. SPARC can be used in two ways: either to reproduce a teacher's policy in a supervised fashion or to help an agent discovering an efficient policy by using the teacher to bias the exploration, limiting the errors and only interacting in desired parts of the environment.

As such, SPARC only defines an interaction framework between a teacher and a learner and is agnostic to the learning mechanism: it can be combined with any algorithms used in SL or RL. This research presented in this thesis explored combinations with three types of algorithms: supervised learning using a feed-forward neural network (Chapter 4), reinforcement learning (Chapter 5), and supervised learning using an instance based algorithm (Chapter 6). However SPARC could be combined with a wide range of other algorithms or techniques such as planning.

Similarly to Inverse Reinforcement Learning (Abbeel & Ng, 2004) or other techniques combining RL and LfD (Billard et al., 2008), if used with a reward function, SPARC could go beyond the demonstrated policy and achieve a performance higher than the demonstration. However this aspect has not been evaluated in this work, but is discussed in Section 7.4.2.

3.6 Application to a New Task

SPARC can be applied to a wide range of tasks in HRI and robotics which currently require tele-operation and where the operator selects high level actions from a finite list.

The interaction designers need to make a number of choices: what do the state and action spaces look like, what learning algorithm is used, how fast does the robot need to take actions, and what does the interface with the user look like. All these choices are application dependent and can have significant impact on the success or failure of SPARC in this new environment. As proposed in Sequeira et al. (2016), the state and action spaces can be decided with the help of mock-up studies or discussions with experts. The algorithm needs to be able to learn quickly from the human, to take demonstrations as inputs and to take into account feedback from the user on suggested actions. In addition, other features might be required to speed up the learning. The correction window can be fixed or variable and needs to be tailored to the application. In some cases where time is not critical, but where incorrect actions have serious impact, this correction window could be set to infinite to force a human to validate the action. And finally, the interface needs to allow the teacher to have a view of the environment (if desired, resembling the algorithm's states as argued by Knox et al. 2014 and Sequeira et al. 2016), to select actions, to react to suggested actions and, if desired, to add features to selections to speed up the learning.

For example, when starting from a WoZ application, the action space is already specified, as well as the interface. In that case, adding SPARC requires to specify the state space and implement a way to sense it, select and implement a learning algorithm and update the interface to allow for the proposition and auto-execution of actions.

3.7 Summary

This chapter introduced Supervised Progressively Autonomous Robot Competencies (SPARC), a novel interaction framework to teach agents a policy. This approach is

suited to teach a robot to interact with humans as it validates the principles defined in Section 2.1.2 (appropriateness of action, adaptivity and autonomy). SPARC starts in a similar fashion than WoZ: the teacher selects actions at any time to be executed by the robot. Then, using a learning algorithm, the agent starts to propose actions back to the teacher who can let them be executed after a short time or cancel them if not appropriate. This mechanism combining selections, suggestions and evaluations of actions ensures the appropriateness of the policy as a human expert could have preempted any inappropriate action before their execution. This additionally provides the adaptivity as the teacher can extend the robot behaviour beyond the current policy. Finally, the learning algorithm associated with the auto-executions of actions intends to decrease the human workload once the robot starts to learn; and, when an acceptable policy is reached, the robot is ready to be deployed to interact autonomously if this outcome is desired.

Chapter 4

Study 1:

Comparison Between SPARC and WoZ

Key points:

- Design of an experiment to explore the influence of SPARC on the teacher's workload and task performance compared to an approach based on WoZ.
- Application target replaced by a robot to ensure repeatability of the target behaviour.
- Design of a robot model exhibiting probabilistic behaviour (simulating a child) with a non-trivial optimal interaction policy.
- Results from a within subject study involving 10 participants show that SPARC achieves a similar performance than WoZ while requiring a lower workload from the teacher.

Parts of the work presented in this chapter have been published verbatim in Senft et al. (2015b). The final publication is available from Springer:

- SPARC: Supervised Progressively Autonomous Robot Competencies. In International Conference on Social Robotics¹.

Technical contribution in this chapter: the author used software from the DREAM project for the touchscreen and the robot functionalities. The author contributed to the material used within the robot control and the Graphical User Interface. Algorithm used from the OPENCV neural network library.

¹http://dx.doi.org/10.1007/978-3-319-25554-5_60

4.1 Motivation

Supervised Progressively Autonomous Robot Competencies (SPARC) has been designed to enable end-users without expertise in computer science to teach a robot a policy while interacting in a sensitive environment, such as Human-Robot Interaction (HRI). By using machine learning and Supervised Autonomy, SPARC intends to allow a field expert to progressively transfer their knowledge to an autonomous agent without having to enforce each action manually. Additionally, as the agent is interacting in the target environment, displaying an appropriate policy, the time spent to teach it is not lost but used to deliver the desired interaction even during the learning phase. For example, in the context of Robot Assisted Therapy (RAT), a therapist would teach the robot during therapy sessions. And, as the therapist is in total control of the robot's action, the behaviour expressed by the robot always fits the desired goals for the therapy. This ensures that even the sessions used to teach the robot have a therapeutic value for the patient involved in the therapy.

SPARC, as a principle, allows to start a robotic application in a Wizard-of-Oz (WoZ) fashion and then move away from it as the robot gains autonomy. The aim of SPARC is twofold: maintaining a high level of performance in the target application while reducing the workload on the teacher over time. As the robot learns, the policy is refined until reaching a point where the robot can be autonomous or only requires a minimal supervision to interact successfully.

The first step to evaluate SPARC was to focus on the teaching interaction, the relation between the robot and its teacher. To evaluate this aspect of the interaction, we decided to select a scenario inspired from RAT for children with Autism Spectrum Disorder (ASD). A child has to complete a task (here emotion recognition) with a robot controlled by a therapist using WoZ (e.g. Vanderborght et al. 2012).

As explained in Chapter 3 and shown in Figure 3.1, applying SPARC to this type of scenario results in two interactions: the teaching interaction (between the teacher and the robot) and the application (between the child and the robot). These two dependent interactions add complexity to the evaluation of the approach and decrease the repeatability of the test benchmark, especially as both humans are impacting each other.

Consequently, we decided to replace the child involved in the therapy by a robot running a model of a child. The setup ends up with two robots interacting together: the *child-robot* completing a therapeutic task and the *wizarded-robot* (controlled by a participant taking the role of the teacher) supporting the child-robot in its task completion (cf. Figure 4.1). Actions from the wizarded-robot impact the child-robot's behaviour and to achieve a high performance in the task, the child-robot needs to receive an efficient supporting policy from the wizarded-robot. As such, the child-robot's performance is used as a proxy to evaluate the performance of the participant in the supervision. This environment with a single human-robot interaction allows us to observe and evaluate the impact of SPARC on the teaching interaction, while controlling for variable interactions with the target.

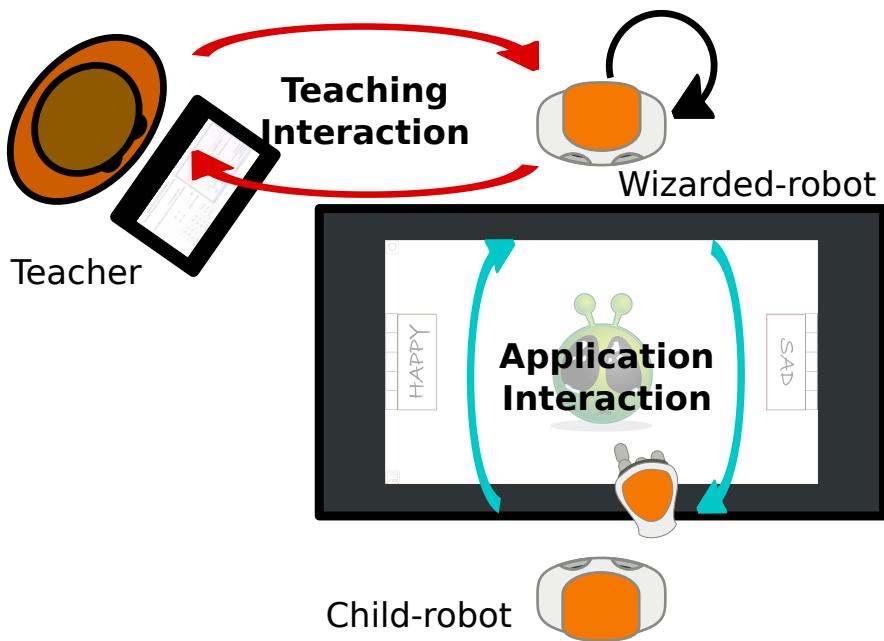


Figure 4.1: Setup of the interaction. The application target, a child has been replaced by a robot for repeatability.

4.2 Scope of the Study

The study presented in this chapter intends to evaluate if the learning component of SPARC could allow participants to teach an efficient policy for a robot interacting with humans. For repeatability concerns, the application target (the child interacting with the robot) was modelled by another robot with a non-deterministic but repeatable behaviour. The control condition is a variation of WoZ, where participant still control a robot but without the learning component. By combining learning and Supervised Autonomy,

4.3. METHODOLOGY

SPARC aims to allow the teacher to maintain a high performance during the interaction while reducing the workload on the teacher over time.

To evaluate the validity of SPARC and the influence of such an approach, four hypotheses were devised:

- H1 The child-robot's performance is a good proxy for the teacher's performance (there is high correlation between the child-robot performance and the teacher performance - maintaining the child-robot internal states high).
- H2 When interacting with a new system, humans progressively build a personal strategy that they will use in subsequent interactions (resulting in variable teacher's outputs and improving their performance in successive interactions).
- H3 Reducing the number of interventions required from a teacher reduces their perceived workload.
- H4 SPARC allows the teacher to achieve similar performance than WoZ but with a lower workload.

H1 represents a validation of the model, ensuring that the child-robot performance represents the efficiency of the policy applied by the teacher. H2 tests that human teachers are not static entities, they adapt their teaching target and their interaction strategy based on their knowledge and understanding of the task. H3 tests one of the motivations behind SPARC: does reducing the number of physical actions from a human to control a robot while requiring the teacher to monitor the robot suggestions lead to a lower perceived workload. Finally, H4 is the main hypothesis, it tests if SPARC can enable a robot to learn a useful policy: reducing the teacher's workload while maintaining a high performance in the application interaction.

4.3 Methodology

4.3.1 Participants

The study involved 10 participants (7M/3F, age $M=29.3$, 21 to 44, $SD=4.8$ years). While SPARC is expected to be usable by anyone, regardless of their knowledge of computer sciences, this first study involved members of a robotic research group assuming the role of the robot supervisor. This decision is supported by the fact that in RAT scenarios, the wizard is typically a technically competent person with significant training controlling

this robot for this therapy. As such, as the participants come from a population expected to assume this type role, the results of the study maintain their applicability to HRI.

4.3.2 Task

This study is based on a real scenario for RAT for children with ASD based on the Applied Behaviour Analysis therapy framework (Cooper et al., 2007). The aim of the therapy is to help a child to develop/practice their social skills by completing tasks with a human or robotic partner. The child has to complete an emotion recognition task by playing a categorisation game with a robot on a mediating touchscreen device (Baxter et al., 2012). The robot can provide feedback and prompts to encourage the child and help them to classify emotions. In the task, images of faces or drawings are shown to the child on the touchscreen, and the child has to categorise them by moving them to one side of the screen or the other depending on whether the picture shown denotes happiness or sadness. In real therapies, the robot would be generally remote-controlled by an operator using WoZ (Riek, 2012).

This study explores if SPARC can be used to teach the robot a correct policy to support the child in this therapy scenario. As timing in human-robot interactions is complex, we have simplified the interaction by making timing discrete, which means the robot now executes actions at a set pace. During these action steps, the selection of an action is decided by following the principles defining the Supervised Autonomy:

1. The robot suggests an action to the teacher.
2. The teacher can select an action for the robot to execute or let the proposed action be executed after a short delay.
3. The robot executes the selected action.
4. Both the robot and the teacher observe the outcome of the action until the next action selection step.

This study compares two conditions: the SPARC condition, where the robot learns from the participant's selections and the WoZ condition where the robot is simply controlled by the participant. As this was the first version evaluating SPARC, the time component of the interaction was artificially discretised; however, this limitation was addressed in future work (cf. Chapter 6). In both conditions, actions can only be executed in

predefined time windows dictated by the dynamics of the interaction (cf. Section 4.3.4). As some situations would require the robot not to act, a ‘wait’ action was added to the SPARC condition and presented an active choice for the participants. In a real WoZ scenario, participants would have to enforce every single action made by the robot, however, a ‘wait’ action would simply correspond to the absence of selection from the wizard. To maintain comparability between the conditions, in the WoZ condition, the robot proposes random actions. While not being a classical WoZ situation, we would expect that the requirement for the teacher to select the ‘wait’ action when required would balance the cases when the robot randomly proposes a correct action. This random behaviour is also the starting point of the SPARC condition.

As mentioned earlier, the focus of the study being on the teaching interaction (the relation between the teacher and the robot), the second interaction (the application) was kept constant by replacing the child by a robot. A minimal model of child behaviour is therefore used to stand in for a real child. A second robot is employed in the interaction to embody this child model: we term this robot the *child-robot* while the robot being directly supervised by the human teacher is the *wizarded-robot* (cf. Figure 4.2).

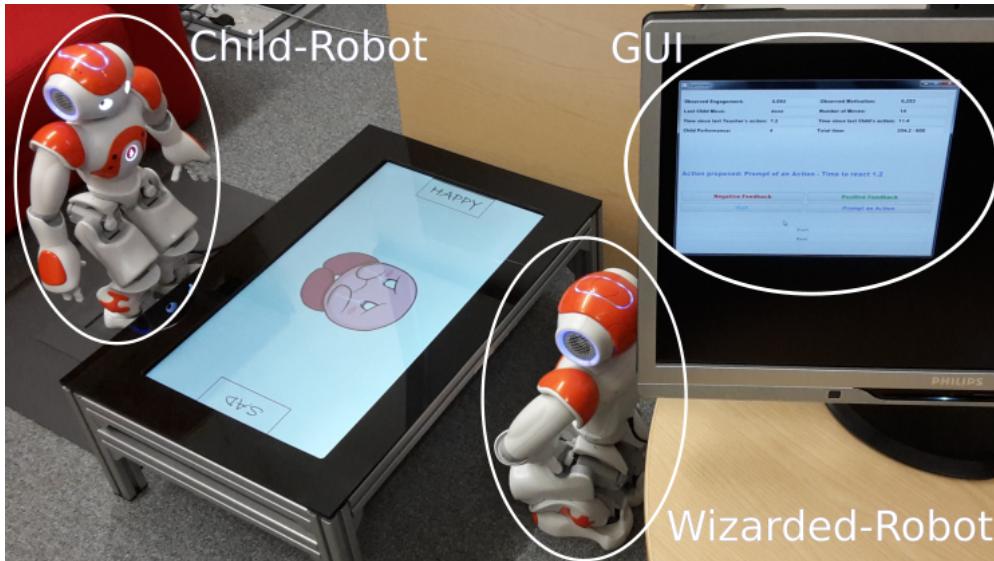


Figure 4.2: Setup used for the user study from the perspective of the human teacher. The child-robot (left) stands across the touchscreen (centre-left) from the wizarded-robot (centre-right). The teacher can oversee the actions of the wizarded-robot through the Graphical User Interface (GUI) and intervene if necessary (right).

4.3.3 Child Model

The purpose of the child model is not to realistically model a child (with or without autism), but to provide a means of expressing some characteristics of the behaviours

we observed in interactions with children in a repeatable manner. The child-robot possesses an internal model encompassing an engagement level and a motivation level. Together these form the state of the child. The engagement represents the involvement of the child in the task, i.e. how often the child-robot will make categorisation moves. And the motivation relates to the seriousness of the child in solving task; in the model, the motivation gives the probability of success of each categorisation move.

These states are bound to the range [-1, 1] and influenced by the behaviour of the wizarded-robot. Values of 1 indicate that the child-robot's behaviour is positive, it is involved in the task. Values of -1 show that the child-robot is actively refusing to participate. And a 0 represents a neutral state where the child-robot is neither especially involved nor actively disengaged. To represent a tendency to return to a neutral state of mild engagement, both states asymptotically decay to zero with no actions from the wizarded-robot. These two states are not directly accessed by either the teacher or the wizarded-robot, but can be observed through behaviour expressed by the child-robot: low engagement will make the robot look away from the touchscreen, and the speed of the categorisation moves is related to the motivation (to which gaussian noise was added). There is thus incomplete/unreliable information available to both the wizarded-robot and the teacher.

As explained in Section 4.3.4, the wizarded-robot's actions impact the child-robot state: congruent actions will tend to increase engagement and motivation. However, if repeated, actions can lead to frustration for the child-robot. If a state is already high and an action from the wizarded-robot should increase it further, there is a chance that this level will sharply decrease. When this happens, the child-robot will indicate this frustration verbally by uttering one of eight predefined strings. This mechanism prevent the optimal strategy to be straightforward: always making actions aiming to increase motivation or engagement. The optimal strategy combines feedback actions and waiting ones to maintain the state values high but prevent them from overshooting. This non-trivial optimal policy approximates better a real human-robot interaction scenario requiring a more complex strategy to be expressed by the robot.

4.3.4 Wizarded-Robot Control

The wizarded-robot is controlled through a GUI (shown in Figure 4.3) and has access to the variables defining the state of the interaction used by the learning algorithm:

4.3. METHODOLOGY

- Observed engagement.
- Observed motivation.
- Type of last categorisation made by the child-robot (good/bad/done).

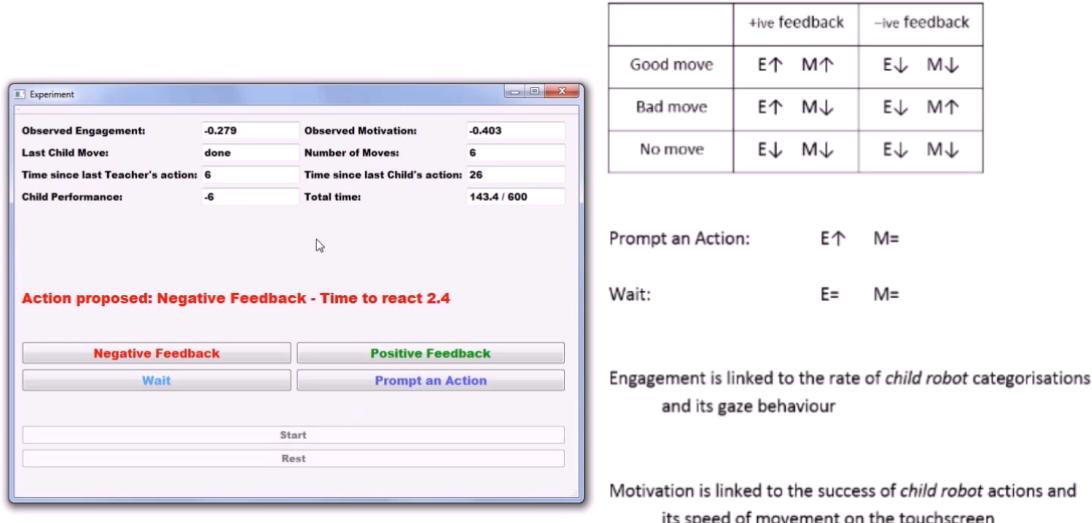


Figure 4.3: Screenshot of the interface used by the participants, the GUI on the left allows to control the robot and a summary of the actions' impact is displayed on the right.

Additionally, other metrics are displayed to the teacher but not used by the algorithm:

- Number of categorisations made by the child-robot.
- Time since teacher's last action.
- Time since child-robot's last action.
- Child-robot's performance.
- Total time elapsed.

The wizarded-robot has a set of four actions it can execute, each represented by a button on the GUI:

- **Prompt an Action:** Encourage the child-robot to do an action.
- **Positive Feedback:** Congratulate the child-robot on making a good classification.
- **Negative Feedback:** Supportive feedback for an incorrect classification.
- **Wait:** Do nothing for this action opportunity, wait for the next one.

The impact of actions on the child-robot depends on the internal state and the type of the last child-robot move: good, bad, or done (meaning that feedback has already been given for the last move and supplementary feedback is not necessary). A *prompt* increases the engagement, a *wait* has no effect on the child-robot's state, and the impact of positive and negative feedback depends on the previous child-robot move. Congruous feedback (positive feedback for correct moves; negative feedback for incorrect moves) results in an increase in motivation, but incongruous feedback can decrease both the motivation and the engagement of the child-robot. The teacher therefore has to use congruous feedback and prompts.

However, as mentioned in Section 4.3.3, if the engagement or the motivation exceeds a threshold, their value can decrease abruptly to simulate the child-robot being frustrated. This implies that the optimal policy consist on providing congruous feedback and prompts, but also requires wait actions to prevent the child-robot from becoming frustrated and maintain its state-values close to the threshold without exceeding it. However, as the participants do not know the parameters of the models running on the child-robot and do not have access to the exact state values, this optimal policy is not available to participants. A ‘good’ strategy keeping the engagement and motivation high leads to an increase in performance of the child-robot in the categorisation task.

As mentioned earlier, in this study the teacher cannot select actions for the wizarded-robot at any time. Actions can only be executed at specific times triggered by the wizarded-robot: two seconds after each child-robot categorisation or if nothing happened for five seconds since the last wizarded-robot’s action. When these selection windows are hit, the wizarded-robot proposes an action to the teacher by displaying the action’s name and a countdown before execution. The teacher can only select an action in reaction to a proposition from the wizarded-robot; alternatively, if the teacher does nothing in the three seconds following the suggestion, the action proposed by the wizarded-robot is executed. This mechanism allows the teacher to passively accept a suggestion or actively intervene by selecting a different action and forcing the wizarded-robot to execute it.

4.3.5 Learning Algorithm

In the SPARC condition, the robot learns to reproduce the policy displayed by the teacher. For this study, the algorithm is of incidental importance, any algorithm could

have been used, but as only supervised learning is required, we used a Multi-Layer Perceptron (MLP) with five input nodes: one for the observed motivation, one for the observed engagement and three binary (+1/-1) inputs for the type of the previous move: good, bad, or done. The hidden layer had six nodes and the output layer four: one for each action and each layer used a sigmoid activation function. The suggested action is selected by applying a Winner-Take-All strategy on the value of the output node and then displayed on the GUI before execution. The network is trained with back propagation: after each new decision from the teacher a new training point is added with the selected action node having +1 while the others are set to -1. The network is fully retrained with all the previous state-action pairs and the new one between each selection step.

This learning algorithm, MLP, is not optimal for online learning for real time interactions as the learning should happen quickly between learning iterations. However, as the length of interaction (and so the number of datapoints) is limited, the network can be retrained between consecutive uses. Finally, the desired learning behaviour being purely supervised learning, this type of algorithm has been deemed suitable for this study.

4.3.6 Interaction Protocol

The study compared two conditions: a learning robot adapting its propositions to its user (the SPARC condition) and a non-learning robot constantly proposing random actions (the WoZ condition). The child-robot controller was kept constant in both conditions, while the state was reset between interactions. The study used a within subject design with balancing of order: each participant interacted with both conditions, and the order of interaction has been balanced between participants to control for any ordering effects. In the order S-W the participants first interact with the learning wizarded-robot in the SPARC condition, and then with the non-learning one in the WoZ condition; and in the order W-S, this interaction order is inverted (starting with WoZ then SPARC). Participants were randomly assigned to one of the two orders.

The interactions took place on a university campus in a dedicated experiment room. Both robots were Aldebaran Nao, one of which had a label indicating that it was the child-robot. The robots faced each other with a touchscreen between them. The participant, assuming the role of the teacher, sat at a desk to the side of the wizarded-robot, with a screen and a mouse to interact with the wizarded-robot (fig. 4.2). Participants were able to see the screen and the child-robot.

A document explaining the interaction scenario was provided to participants with a demographic questionnaire. After the information was read, a 30s video presenting the GUI in use was shown to participants to familiarise them with the interface, without biasing them towards any particular control strategy. Then participants clicked a button to start the first interaction which lasted for 10 minutes. The experimenter was sat in the room outside of the participants' field of view. After the end of the first interaction, a post-interaction questionnaire was administered. Similarly, in the second part of the experiment, the participants interacted with the other condition and completed a second post-interaction questionnaire. Finally, a post-experiment questionnaire asked participants to explicitly compare the two conditions. All questionnaires and the information sheet are presented in Appendix D, and the files and the instruction video showing the interface can be found online².

4.3.7 Metrics

Two types of metrics have been recorded for this study: interaction data representing objective behaviours and the performance of the participants and subjective data through questionnaires.

4.3.7.1 Interaction Data

The state of the child-robot and the interaction values were logged at each step of the interaction (at 5Hz). At each selection step, all of the human actions were recorded: acceptance of the wizarded-robot's suggestion, auto-execution and selection of another action (intervention) as well as the current state of the child-robot (motivation, engagement and performance).

The first metric is the performance achieved by participants in each interaction. As the policy applied by the participants cannot be evaluated directly, the performance of the child-robot in the task (number of correct categorisations minus number of incorrect categorisations) is used as a proxy for the participant performance. H1 evaluates if this approximation is valid by analysing the relation between the performance of the child-robot and the value of its inner states. If a strong correlation is found, it would demonstrate that a good supervision policy (managing to keep the engagement and the motivation of the child-robot high) leads to a high performance. As such, this child-robot

²<https://emmanuel-senft.github.io/experiment-woz.html>

performance represents how efficient the policy executed by the wizarded-robot was when controlled by a participant.

The second important metric is the intervention ratio: the number of times a user chooses a different action than the one proposed by the wizarded-robot, divided by the total number of executed actions. This metric represents how often on average a user had to correct the robot and could be related to the workload the user had to face to control the robot.

4.3.7.2 Questionnaire Data

Participants answered four questionnaires: a demographic one before the interaction, two post-interaction ones where they were asked to evaluate the last interaction with the robots and a post-experiment questionnaire where they had to compare the two conditions. All the rating questionnaires used seven item Likert scale. For clarity for participants, in the questionnaires the wizarded-robot is named ‘teacher-robot’ (as it was ‘teaching’ the child-robot).

Post-Interaction questions:

- The child-robot learned during the interaction (Likert scale 1 to 7).
- The performance of the child-robot improved in response to the teacher-robot’s actions (Likert scale 1 to 7).
- The teacher-robot is capable of making appropriate action decisions in future interactions without supervision (Likert scale 1 to 7).
- The teacher-robot always suggested an incorrect or inappropriate actions (Likert scale 1 to 7).
- By the end of the interaction, my workload was very light (Likert scale 1 to 7).
- What did you pay most attention during the interaction? (child-robot, touchscreen, GUI, other).

Post-experiment questions:

- There was a clear difference in behaviour between the two teacher-robots (Likert scale 1 to 7).

- There was a clear difference in behaviour between the two child-robots (Likert scale 1 to 7).
- Which teacher-robot was better able to perform the task? (first, second).
- Which teacher-robot did you prefer supervising? (first, second).

4.4 Results

Graphs presented in this chapter and throughout this thesis have been generated using the seaborn package for Python and matplotlib (Waskom et al., 2017). Data is usually represented using violin plots, a graphical representation featuring the probability density of the data. In addition to this plot, graph also display the raw data points and in Chapters 4 and 5 a minimal box plot showing the quartiles and outliers.

4.4.1 Interaction Data

Figure 4.4 presents the aggregated results (collapsed between orders) for the performance and the final intervention ratio for both conditions. While the number of participants is not sufficient to compute descriptive statistics, overall interaction results seemed to show that both conditions lead to similar performance (SPARC: 32.6 (95% CI [27.89,37.31]) - WoZ: 31.4 (95% CI [25.9,36.9])). As supported by the absence of overlap between the 95% CI, the SPARC condition tended to require less interventions (intervention ratio: SPARC: 0.38 (95% CI [0.29,0.47]) - WoZ: 0.59 (95% CI [0.52,0.67])).

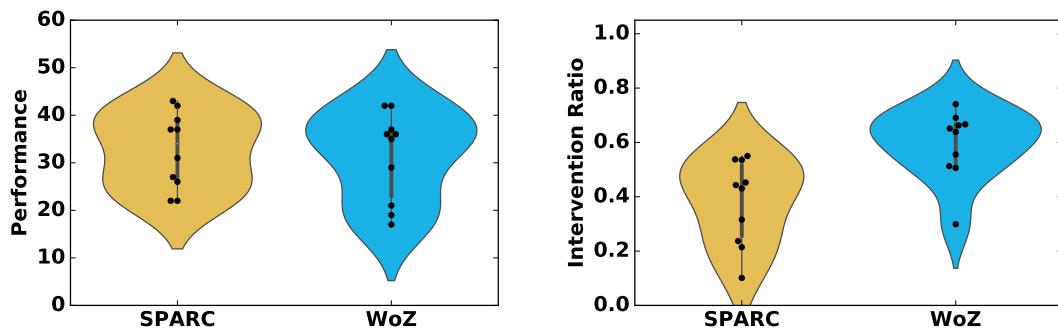
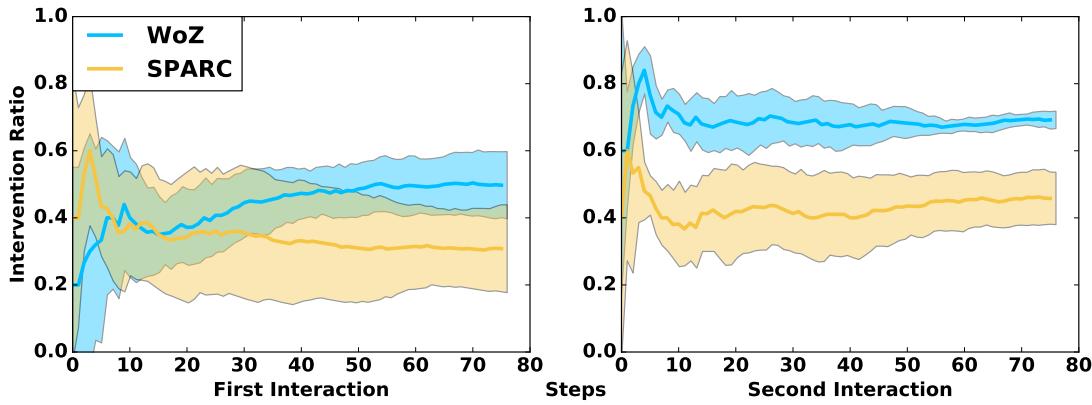


Figure 4.4: Aggregated comparison of performance and final intervention ratio for both conditions. Dots represent individual datapoint (N=10 per condition) and shaded area the probability distribution most likely to lead to these points.

Figure 4.5 presents the evolution of intervention ratio for each condition, and their order. During the first interaction, participants discovered the interface and how to interact with it, which resulted in a high variation of the intervention ratio in the first 20 steps (with one step corresponding to one proposition of action from the wizarded-robot). However

4.4. RESULTS

in the second phase of the interaction, when participants had developed their teaching policy, there was a tendency for SPARC to require a lower number of intervention than WoZ. This effect was higher in the second interaction, where as soon as 5 steps, the two conditions differentiated without overlap of the 95% CI of the mean. This would indicate that the two conditions differ in term of required interventions.



*Figure 4.5: Evolution of intervention ratio over time for both conditions and both orders.
Shaded area represents the 95% CI (N=5 for each plot).*

For both the performance and the intervention ratio, a strong ordering effect was observed. Figure 4.6 and Table 4.1 present the performance and final intervention ratio separated by condition and order. In both orders, the performance in the second interaction was higher than the one in the first interaction, as the participants were used to the system and developed an efficient interaction policy. On the other hand, the performance between condition for the same interaction number was similar (in both their first and second interactions, the condition of interaction did not impact the performance). However, for both orders, when comparing between condition for the same interaction number, the intervention ratio was lower when using SPARC compared to WoZ. This indicates that when the wizarded-robot learned using SPARC, a similar performance was attained as with WoZ, but the number of interventions required to achieve this performance was lower.

Additionally, a strong positive correlation (Pearson's $r=0.79$) was found between the average child-robot motivation and engagement and its performance which shows that the performance achieved by the child-robot represented the capacity of the teacher to keep both engagement and motivation high.

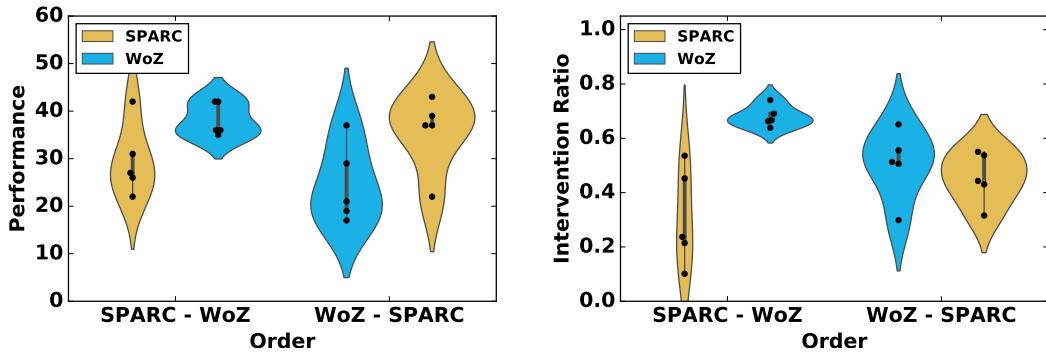


Figure 4.6: Performance achieved and final intervention ratio separated by order and condition. For each order, the left part presents the metric in the first interaction (with one condition) and the right part the performance in the second interaction (with the other condition).

Table 4.1: Average performance and intervention ratio separated by condition and order.

	Order S-W		Order W-S	
	SPARC (int 1)	WoZ (int 2)	WoZ (int 1)	SPARC (int 2)
Performance M	29.6	38.2	24.6	35.6
95% CI	[23.6,35.6]	[35.5,40.9]	[18.1,31.1]	[29.3,41.9]
Intervention Ratio M	0.31	0.68	0.5	0.46
95% CI	[0.17,0.45]	[0.65,0.71]	[0.4,0.61]	[0.38,0.53]

4.4.2 Questionnaire Data

The post-interaction questionnaires evaluated the participant's perception of the child-robot's learning and performance, the quality of suggestions made by the wizarded-robot, and the experienced workload. All responses used seven point Likert scales. Table 4.2 presents separated results for the questions asked in the post-interaction questionnaires, with more details for the questions exhibiting differences in Figure 4.7.

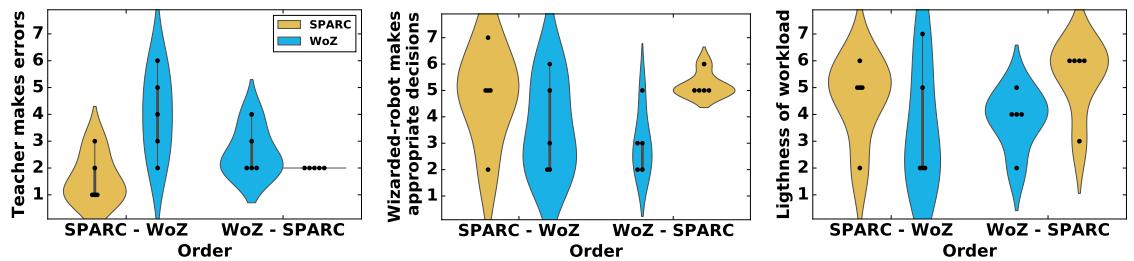


Figure 4.7: Questionnaires results on robot making errors, making appropriate decisions and on lightness of workload.

Across the four possible interactions, the rating of the child-robot's learning was similar ($M=5.25$, 95% CI [4.8, 5.7]). As the child-robot was using the same interaction model in

Table 4.2: Average reporting on questionnaires separated by condition and order.

	Order S-W		Order W-S	
	SPARC (int 1)	WoZ (int 2)	WoZ (int 1)	SPARC (int 2)
Child learns M	5.2	5.2	5.2	5.4
95% CI	[3.7,6.7]	[3.8,6.6]	[4.2,6.2]	[4.7,6.1]
Child's performance M	4.6	5.0	5.0	4.4
95% CI	[3.4,5.8]	[3.3,6.8]	[4.0,6.0]	[3.7,5.1]
Wizarded-robot makes errors M	1.6	4.0	2.6	2.0
95% CI	[0.9,2.3]	[2.8,5.2]	[1.9,3.3]	[2.0,2.0]
Wizarded-robot makes appropriate decisions M 95% CI	4.8 [3.4,6.2]	3.6 [2.2,5.0]	3.0 [2.0,4.0]	5.2 [4.9,5.6]
Lightness of workload M	4.6	3.6	3.8	5.4
95% CI	[3.4,5.8]	[1.8,5.4]	[2.9,4.7]	[4.4,6.5]

all four conditions, this result is expected. There is a slight tendency to rate the child's performance as being higher in the WoZ condition but the error margin is too high to conclude anything.

Participants rated the wizarded-robot as more suited to operate unsupervised with SPARC than with WoZ (95% Confidence Interval of the Difference of the Mean (CIDM) for S-W ordering [-0.2, 2.6], CIDM for the W-S ordering [1.6, 2.8]).

Similarly, a trend was found showing that the wizarded-robot with SPARC is perceived as making fewer errors than with WoZ (CIDM for S-W ordering [1.3, 3.4], CIDM for the W-S ordering [0.1, 1.1]).

Also participants tended to rate the workload as lighter when interacting with SPARC, and this effect is much more prominent when the participants interacted with the WoZ first (CIDM for S-W ordering [-0.6, 2.6], CIDM for the W-S ordering [0.7, 2.5]).

Most of the difference of mean interval exclude 0 or include it marginally, which would indicate tendency of difference, but due to the low number of participants, no statistical tests are applicable and as such no significance can be demonstrated.

4.5 Discussion

Strong support for H1 (a good teacher leads to a better child performance) was found, a correlation between the average value of states (engagement and motivation) and the

final performance for all of the 10 participants was observed ($r=0.79$). This validity check confirms that the performance of the child robot reflects the performance of the teacher in this task: supervising the wizarded-robot to execute an efficient policy maximising the inner state of the child-robot. Additionally, the model of the child robot exhibited the desired behaviour: allowing a wide range of performances without one obvious optimal policy.

The results also provided support for H2 (teachers create personal strategies): all the participants performed better in the second interaction than in the first one. This suggests that participants developed a strategy when interacting with the system in the first interaction, and were able to use it to increase their performance in the second interaction. Looking in more detail at the interaction logs, different strategies for the wizarded-robot can be observed. For instance, the ratio of waiting action compared to other supportive actions varied between participants (from 4.7% to 56%). This variation demonstrates that despite the repeatability of the environment, different participants applied different strategies when controlling and teaching a robot.

H3 (reducing the number of interventions reduces the perceived workload) is partially supported: the results show a trend for participants to rate the workload as lighter when interacting with SPARC, and another trend between using SPARC and the intervention ratio. However, when computing the correlation between the intervention ratio and the reported workload, a strong effect can only be observed in the second interaction ($\rho = -.622$). In the first interaction, the main cause of the workload is probably the discovery of the system and how to interact with it rather than the requirement to manually select actions for the robot. Nevertheless, regardless of the order of the interactions, SPARC tended to receive higher ratings for lightness of workload and required fewer interventions to be controlled which indicates that using SPARC could decrease the workload on robots' supervisors compared to WoZ.

Finally, H4 (using learning maintains similar performance, but decreases the workload) is supported: interacting with a learning robot in the SPARC condition resulted in a similar performance than interacting with a non-learning robot in the WoZ condition, whilst requiring fewer active interventions from the supervisor and a lower workload to control. Reducing the workload on the robot operator has real world utility, for example, in the context of RAT, it might free time for the supervisor to allow them to focus on other

aspects of the intervention, such as analysing the child's behaviour rather than solely controlling the robot.

It should be noted that the actual learning algorithm used in this study is only of incidental importance, and that certain features of the supervisor's strategies may be better approximated with alternative methods – of importance for the present work is the presence of learning at all. Other algorithms and ways to handle time have been used in the following studies presented in Chapters 5 and 6.

4.6 Summary

Using a suggestion/intervention system, SPARC allowed online learning for interactive scenarios, thus increasing autonomy and reducing the demands on the supervisor. Results showed that the learning component of SPARC allowed participants to achieve a similar performance as interacting with a non-learning robot, but requiring fewer interventions to attain this result. This suggests that while both conditions allowed the participants to reach a good performance, with SPARC, the presence of learning shifts part of the burden of selecting actions onto the wizarded-robot rather than on the human. Using SPARC, the robot partially learnt an interaction policy which decreased the requirement on the teacher to physically enforce each robot's action. This indicates that a learning robot could reduce the workload on the operator freeing them to do more valuable tasks and that SPARC could be an efficient interaction framework to operate this learning. In addition to providing a robot with autonomy, this reduction of workload has real world implications, in the context of RAT, it could allow the therapist to focus more on the child than on the robot, with improved therapeutic outcomes as potential result.

Chapter 5

Study 2:

Importance of Control Over the Learner

Key points:

- Design of an experiment comparing SPARC and another Interactive Machine Learning method: Interactive Reinforcement Learning (IRL).
- The application domain is a replication of the world used in early studies evaluating IRL.
- IRL uses partial guidance to the robot and explicit rewarding of the robot's action to teach it a policy.
- SPARC uses full control over the robot's action, implicit rewards and evaluation of intentions rather than actions.
- Results from a mixed design study involving 40 naive participants show that SPARC achieves a better performance and an easier and faster teaching than IRL.

Parts of the work presented in this chapter have been published verbatim in Senft et al. (2017a). The final publication is available from Elsevier:

- Supervised Autonomy for Online Learning in Human-Robot Interaction. In Pattern Recognition Letters¹.

Technical contribution in this chapter: the author reimplemented every part of the original system using Qt.

¹<https://doi.org/10.1016/j.patrec.2017.03.015>

5.1 Motivation

As described in Chapter 2, previous work in Interactive Machine Learning (IML) showed that humans want to teach robots not only with feedback on their actions but also by communicating the robots what they should do (Thomaz & Breazeal, 2008; Amershi et al., 2014). However, in most research where agents are taught policies using human guidance, the teacher is given little or no control over the agent's actions and has to observe the agent executing an action even when knowing that this action is incorrect (cf. Section 2.3.4). This chapter explores how these IML approaches could be improved by applying the principles of Supervised Progressively Autonomous Robot Competencies (SPARC) defined in Chapter 3. This chapter also presents experimental results demonstrating how these principles influence the learning process, the agent performance and the user experience and how these results compare to other traditional IML approaches.

The study presented in Chapter 4 explored how SPARC could be used with Supervised Learning to replicate a teacher's policy. However, some of the most promising features of IML arise when combined with Reinforcement Learning (RL) as it might allow an agent to learn beyond the demonstrations (Abbeel & Ng, 2004). As such, this chapter proposes a way to apply the principles underlying SPARC to classical feedback based RL and evaluates how this human control over the robot's actions impacts the learning. This chapter presents results from a study involving 40 participants comparing the teaching efficiency and user experience of SPARC to Interactive Reinforcement Learning (IRL), another IML approach offering less control but having been validated in previous studies (Thomaz & Breazeal, 2008).

5.2 Scope of the Study

5.2.1 Interactive Reinforcement Learning

IRL implements the principles presented in Thomaz & Breazeal (2008) (hereafter the 'original study' or 'original paper'): a human supervises and teaches an agent to interact autonomously in an environment. This teaching is achieved by providing guidance and positive or negative feedback on the last action executed by a robot. In this study, the algorithm controlling the robot combines these human feedback with environmental ones to form a reward used to update a Q-table. This Q-table assign a Q-value (interest

of taking an action) to every state-action pair and is used to select the next action. Three additions to the standard interaction mechanism have been proposed and implemented by Thomaz and Breazeal and are used in this study as well: guidance, communication by the robot and an undo mechanism (Thomaz & Breazeal, 2008).

The guidance channel emerged from the results of a pilot study where participants assigned rewards to objects to indicate that the robot should do something with them. With the guidance channel, teachers can direct the attention of the robot toward certain items in the environment, informing the robot that it should use them in its next action. This guidance behaviour offers partial control over the robot's actions restricting the executable options, but cannot be used to explicitly set the robot's behaviour.

Additionally, the robot communicates uncertainty by directing its gaze toward different parts of the environment with equally high probabilities of being used next. The aim of this communication of uncertainty is to provide transparency about the robot's internal state, for example indicating when the robot is unsure about its next action and that guidance should be provided.

Finally, the undo mechanism aims to provide a way for the teacher 'cancel' the robot's action, to bring it back to relevant part of the world after an error, in order to speed up the teaching. After receiving a negative reward, the robot tries to cancel the effect of the previous action on the environment (if possible), resulting in an undo behaviour. As shown in the original studies, these three additions improve the robot's performance on the task and the user experience.

In summary, with IRL, teachers have two ways to transmit information to the robot: a reward channel (providing a numerical evaluation of the last action) and a guidance channel (directing the robot's attention toward parts of the state to restrict the exploration).

5.2.2 SPARC

SPARC uses a single type of input from the human similar to the guidance in IRL and no reward channel. However with SPARC, the guidance channel directly controls the actions of the robot. On the other hand, the robot communicates all of its intentions (i.e the action it plans to execute next) to its teacher by looking at the corresponding part of the environment. Following the principles proposed in Section 3.3, the teacher can either not intervene, letting the robot execute the suggested action or step in and

force the robot to execute an alternative action. This combination of suggestions and corrections gives the teacher's full control over the actions executed by the robot. This also makes the rewards redundant. Rather than requiring the human to explicitly provide rewards, a positive reward is directly assigned to each action executed by the robot as it has been either enforced or passively approved by the teacher.

5.2.3 Differences Between IRL and SPARC

Unlike IRL, SPARC offers the user full control over the actions executed by the robot. SPARC changes the learning paradigm from learning from the human's evaluation of actions' impacts to learning from the human's knowledge and the *expected* impact of actions. An expert in the task domain evaluates the appropriateness of actions before their execution and can guide the robot to act in a safe and useful manner. This implies that the robot does not rely on observing the negative effects of an action to learn to avoid it (as with IRL or more generally RL), but rather it learns what the best action is for each state. Even in a non-deterministic environment such as human-robot interactions, some actions can be expected to have a negative consequence, and the human teacher should be able to stop the robot from ever executing them, preventing the robot from causing harm to itself or its social or physical environment.

Another noticeable difference is the type of information the robot communicates with the user: in IRL, the robot communicates its uncertainty about an action and with SPARC its unambiguous intention to execute an action. Similarly, the communication from the user to the robot differs between the two approaches. In SPARC the user can offer the whole action space as commands to the robot, which removes the need for explicit rewards, while in IRL, the teacher can guide the robot toward a subset of the action space and has to manually provide feedback to evaluate the robot's decisions. A result is that the quantity of information provided by the user to the robot is similar for both IRL and SPARC.

5.2.4 Hypotheses

Three hypotheses have been tested in the study:

H1 *Effectiveness and efficiency with non-experts.* SPARC will lead to better learning than IRL when used by non-experts (higher performance, faster learning, lower

number of inputs used, lower mental effort on the teacher and lower number of errors during the teaching phase).

H2 *Safety with experts.* SPARC can be used by expert users (knowledgeable in the interaction process) to teach a policy safely, quickly and efficiently, achieving better results than other IML methods lacking control.

H3 *Control.* Teachers prefer a method in which they can have more control over the robot's actions.

5.3 Methodology

5.3.1 Participants

A total of 40 participants (age $M=25.6$, $SD=10.09$; 24F/16M) were recruited using a tool provided by the University of Plymouth to reach a mixed population of students and non-student members of the local community². At the start of the experiment, all participants gave written informed consent, were told of the option to withdraw at any point and completed a demographic questionnaire. Participants were mostly not knowledgeable in machine learning and robotics (familiarity with machine learning $M=1.8$, $SD=1.14$; familiarity with social robots $M=1.45$, $SD=0.75$ - Likert scale ranging from 1: not at all familiar to 5: extremely familiar). The study lasted around one hour and followed a 2x2x3 mixed design where participants interacted with both conditions three times. To address ordering effects, the order of interaction was counterbalanced between two groups: group 1 interacting with IRL for three sessions then with SPARC for another three sessions and the interaction order was inverted for group 2 (see also Figure 5.2). Participants were distributed randomly between the two groups whilst balancing gender and age. All participants received remuneration at the standard U.K. living wage rate, pro rata.

In addition to naive non-expert users, an expert user (the author) interacted five times with each system following a strictly optimal strategy for both conditions. These results from the expert are used to evaluate H2 and show the optimal characteristics of each system (IRL and SPARC) when used by trained experts in robot interaction, such as therapists in the context of assistive robotics.

²<https://uopsop.sona-systems.com/Default.aspx?ReturnUrl=%2f>

5.3.2 Task

The task used in this study is the same as Thomaz & Breazeal (2008): “Sophie’s kitchen”, a simulated environment presented on a computer where a virtual robot has to learn how to bake a cake in a kitchen. As the source code was not available, the task was reimplemented to stay as close as possible to the description in the paper and the online version of the task³.

The scenario is the following: a robot, Sophie, is in a kitchen with three different locations (shelf, table and oven) and five objects (flour, tray, eggs, spoon and bowl) (see Figure 5.1a). The participant has to teach Sophie to bake a cake by guiding it through a sequence of steps while giving enough feedback so the robot learns a correct series of actions leading to the completion of the task. There are six crucial steps to achieve a successful result:

1. Put the bowl on the table (Figure 5.1b).
2. Add one ingredient to the bowl (flour or eggs).
3. Add the second ingredient (Figure 5.1c).
4. Mix the ingredients with the spoon to obtain batter (Figure 5.1d).
5. Pour the batter in the tray (Figure 5.1e).
6. Put the tray in the oven (Figure 5.1f).

The environment is a deterministic Markov Decision Process (MDP), defined by a state (the position and state of each object and of the robot), a set of actions (move left, move right, pick up, drop and use), a deterministic transition function and an environmental reward function. The environment includes end states, corresponding to a success or a failure, which reset the simulation to the initial state and provide a reward (+1 for success and -1 for failure). All the other states corresponding to intermediate steps have a reward of -0.04 to penalise long sequences. Different policies can lead to success, but many actions end in a failure state, for example putting the spoon in the oven. This environment includes a large number of possible states (more than 10,000), success and failure states and a sparse environmental reward function. These elements increase

³<http://www.cc.gatech.edu/~athomaz/sophie/WebsiteDeployment/>

the value of having a teacher present to support the learning. As argued by Thomaz and Breazeal in the original paper, this environment provides a good setup for evaluating methods for teaching a robot.

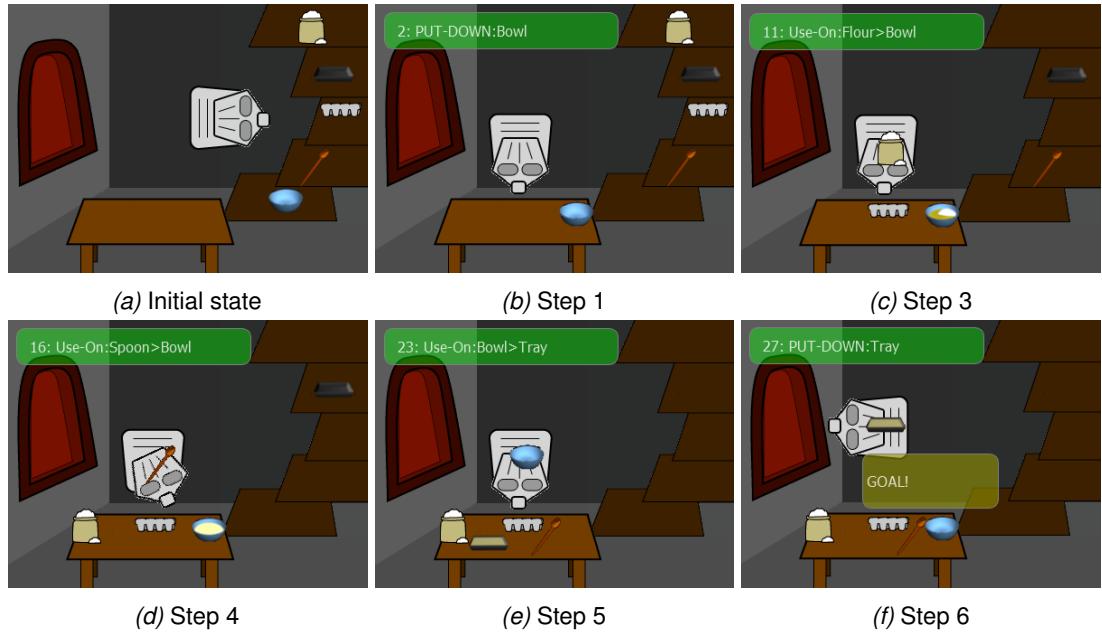


Figure 5.1: Presentation of different steps in the environment. (a) initial state, (b) step 1: bowl on the table, (c) step 3: both ingredients in the bowl, (d) step 4: ingredients mixed to obtain batter, (e) step 5: batter poured in the tray and (f) step 6 (success): tray with batter put in the oven. (Step 2: one ingredient in the bowl has been omitted for clarity, two different ingredients could be put in the bowl to reach this state)

5.3.3 Implementation

Two conditions were constructed to compare the IRL and SPARC approaches on this task. The underlying learning mechanism is identical in both conditions. The only differences lie in the manner of interaction (inputs to and from the algorithm) and the amount of control over the robot's actions. With IRL teachers have to explicitly provide rewards and have a partial control over the action selection, while with SPARC rewards are implicit and the teacher's control over actions is total.

The learning algorithm (see Algorithm 2 and 3) is a variation on Q-learning, without reward propagation⁴. This guarantees that any learning by the robot is due to the human's teaching, and as such provides a lower bound for the robot's performance. Similarly to the original study, the algorithm uses a learning rate $\alpha = 0.3$ and a discount factor $\gamma = 0.75$. Sources are available online⁵.

⁴In Q learning the update function is $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_a Q(s_{t+1}, a)) - Q(s_t, a_t))$

⁵<https://github.com/emmanuel-senft/experiment-irl>

Table 5.1: Simplified outline of algorithms used for both condition.

Algorithm 2: SPARC	Algorithm 3: IRL
<p>inputs : s_t: current state A_t: actions available Q: Q-table</p> <p>outputs : a_t: selected action Q: updated Q-table</p> <p>while learning do</p> <ul style="list-style-type: none"> $a_t = \arg\max_{a \in A_t} Q[s_t, a]$ look at object or location used in a_t while waiting for command (2 seconds) do <ul style="list-style-type: none"> if received command then <ul style="list-style-type: none"> $a_t = \text{command}$ $r_t = 0.5$ else <ul style="list-style-type: none"> $r_t = 0.25$ <p>Act in the world: execute a_t, transition to s_{t+1} $r_t = r_t + r_{\text{environment}}$</p> <p>Learn:</p> $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_a Q(s_t, a) - Q(s_t, a_t))$	<p>inputs : s_{t+1}: current state A_{t+1}: actions available Q: Q-table</p> <p>outputs : a_{t+1}: selected action Q: updated Q-table</p> <p>while learning do</p> <ul style="list-style-type: none"> $A'_{t+1} = [a^1 \dots a^n]$, n actions with high $Q[s_{t+1}, a^i]$ while waiting for guidance and reward on a_t (2 seconds) do <ul style="list-style-type: none"> if $n > 1$ then <ul style="list-style-type: none"> indicate confusion if received reward r'_t then <ul style="list-style-type: none"> $r_t = r_t + r'_t$ if receiving guidance then <ul style="list-style-type: none"> $a_{t+1} = \text{guidance}$ else <ul style="list-style-type: none"> $a_{t+1} = \arg \max_{a \in A_{t+1}} Q[s_t, a]$ <p>Learn (on the previous step):</p> $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_a Q(s_t, a) - Q(s_t, a_t))$ <p>Act in the world: execute a_{t+1}, transition to s_{t+2} $r_{t+1} = r_{\text{environment}}$</p>

As shown in Table 5.1, another difference between the conditions is that with SPARC, the algorithm learns immediately after executing an action (and only with positive rewards). On the other hand, IRL learns about an action just before executing the next one, based on a positive or negative evaluation received between the actions.

5.3.3.1 Interactive Reinforcement Learning

We have implemented IRL following the principles presented in Thomaz & Breazeal (2008). The user can use the left mouse-click to display a slider providing rewards. Guidance is implemented by right-clicking on objects to direct the robot's attention toward a specific object. Guidance can only be provided for objects the robot is facing, otherwise right-clicking has no effect. Following a guidance message, the robot will execute the candidate action involving the object. The action space is not entirely covered by this guidance mechanism: for example, it does not cover moving from one location to another. This guidance gives a partial opportunity to the user to limit the exploration for the current step, without preventing the robot to explore in further steps.

Similarly to the original study, the robot indicates its intention and confusion by *gazing* to objects. In this case, the head of the robot would point towards an object indicating that the robot considers the action related to this object. This gazing behaviour is simple to interpret as objects are located in space in such a way that only one can be faced by the robot at the same time. In the case of confusion, the robot would move its head between different direction indicating the n objects it is considering to use in its next action.

Some modifications to the original study were required due to the lack of implementation details in the original paper, one of them being the use of a purely greedy policy instead of using softmax. As the presence of human rewards and guidance limits the importance of autonomous exploration, the greediness of the algorithm should assist the learning by preventing the robot from exploring outside of the guided policy.

It should be noted that the presence of the human in the learning process alters deeply the concept of convergence. By providing rewards, the teacher can manually force the robot's policy to converge or diverge.

5.3.3.2 SPARC

With SPARC, the robot uses its head gaze toward objects or locations to indicate to the teacher which action the robot is suggesting. This behaviour is comparable to the confusion used in the IRL condition, however, with SPARC, the robot will only face a single object or location to indicate in unambiguous and clear way the action it is planning to do. Similarly to the guidance in IRL, the teacher can use the right click of the mouse on objects to send a ‘command’ to the robot and have it execute the action associated to this object in the current state. However, in this condition, this communication has been extended to also cover locations. With SPARC, the command covers the whole action space: at every time step, the teacher can specify, if desired, the next action to be executed by the robot. Similarly to the guidance, this command can be used on objects only if the robot is facing them. If a robot’s suggested action is not corrected, a positive reward of 0.25 is automatically received (as it has the implicit approval from the teacher). If the teacher selects another action, a reward of 0.5 is given to the selected action (the corrected action is not rewarded). That way, actions actively selected are more reinforced than the ones accepted passively and participants still have access to a wider range of rewards with IRL. This system allows for the use of reinforcement learning with implicit reward assignation, aiming to simplify the teaching interaction.

5.3.4 Interaction Protocol

Participants were divided into two groups and interacted with both IRL and SPARC, with the order of presentation being counterbalanced between groups (see Figure 5.2). Participants in group 1 interacted with IRL first for three sessions and then with SPARC for the three remaining sessions; and the interaction order was inverted for participants in group 2.

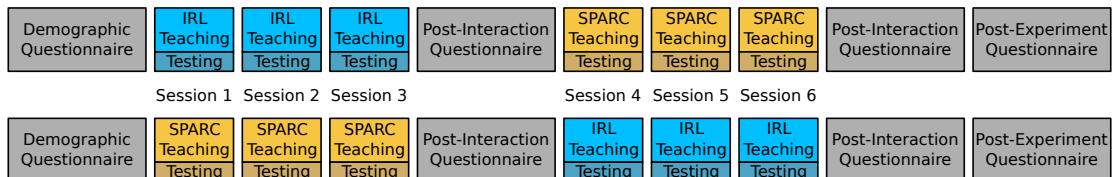


Figure 5.2: A representation of the timeline experienced by participants according to the order they were in. The top row corresponds to group 1 and bottom row to group 2.

After welcoming participants and before interacting with a system, participants completed a demographic questionnaire and received two information sheets. The first one explained the task (describing the environment and how to bake the cake) and the second one described the system they would interact with (IRL or SPARC).

After reading the sheets, participants interacted for three sessions with the system they were assigned to. Each session started with a teaching phase where the participants could interact with the robot to teach it to complete the task. This teaching phase was composed of a number of episodes, corresponding each to a trajectory from the initial state to an end state (success or failure) after which the environment was returned to the initial state. Similarly to the experiment by Thomaz and Breazeal, participants could decide to terminate the teaching phase whenever they desired by clicking on a button labelled ‘Sophie is ready’. But the teaching phase was automatically terminated after 25 minutes to impose an upper time-limit on the study.

After the teaching phase, the robot ran a testing phase where the participant’s inputs, other than a force stop, were disabled. The test stopped as soon as an ending state was reached or the participant forced a stop (e.g. if an infinite loop occurred). This testing phase aimed to evaluate the participants’ performance in the teaching task. The interaction with each system involved three repeated independent sessions with their own teaching and testing phases. This way, we can observe how the interactions evolved as participants got used to interact with a system.

After participants completed their three sessions with the first system, they were asked to complete a first post-interaction questionnaire. Then, they received the information sheet for the second system, interacted with it for three sessions and completed a second post-interaction questionnaire.

At the end of the experiment, participants completed a last questionnaire, the post-experiment questionnaire, received the financial compensation and were explained the goal of the study. All information sheets and questionnaires can be found in Appendix E and questionnaires are described in Section 5.3.5.2.

5.3.5 Metrics

5.3.5.1 Interaction Metrics

We collected four metrics during the teaching phase (teaching performance, teaching time, number of failures and number of inputs provided) and one during the testing phase (the testing performance). All interaction metrics were collected three times per conditions, once for each session. As not all participants reached a success during the testing phases, we used the six key steps defined in Section 5.3.2 as a way to evaluate the performance ranging from 0 (no step was completed) to 6 (the task was successfully completed). For example a testing where the robot put both ingredients in the bowl but reached a failure state before mixing them would have a performance of 3.

The testing performance represents the success of participants in teaching the robot to complete the task. On the other hand, the teaching performance corresponds to the highest step reached by participants in the teaching phase and represents a teaching method's ease of guiding the robot. The teaching time is the duration of the teaching phase, ranging from 0 to 25 minutes. The number of failures is the number of times a participant reached a failure state during the teaching phase. It can be related to the risks involved by the teaching; a safe teaching process should lead to a low number of failures, while a risky one would have a high number of failures. The number of inputs corresponds to the number of commands, guidances or feedback inputs used in a teaching session. Similarly to the teaching time, the number of inputs can be seen as the quantity of efforts invested in the teaching process.

5.3.5.2 Questionnaires

The post-interaction and post-experiment questionnaires provided additional introspective information to compare with the quantitative data from the interaction. Two principal metrics were gathered: the workload on participants and the perception of the robot.

Workload is an important factor when teaching robots. As roboticists, our task is to minimise the workload for the robot's user and to make the interaction as smooth and efficient as possible. Multiple definitions for workload exist and various measures can be found in the literature (e.g. Wierwille & Connor 1983; Moray 2013). Due to its widespread use in human factors research (Hart, 2006) and clear definition and evaluation criteria, we used the NASA-Task Load Index (TLX) (Hart & Staveland, 1988). Following the

methodology proposed to administer the Raw NASA-TLX in Hart (2006), we averaged the values from all 6 scales (mental, physical and temporal demand, performance, effort and frustration) ranging from 0 – low workload – to 20 – high workload – (with the performance being reversed as a high performance relates to a low workload) to obtain a single workload value per participant for each interaction. This assessment was made during the post-interaction questionnaires. This resulted in two measures of workload per participant, one for each condition.

Finally, the participants' perception of the robot was also evaluated in the post-interaction and post-experiment questionnaires using rating questions (measured on a 5 item Likert scale), binary questions (where participants had to select one of the two system), and open questions on the preference of system and the naturalness of the interaction.

5.4 Results

Most of the results collected in the study were non-normally distributed. Both ceiling and floor effects could be observed depending on the conditions and the metrics. For instance, for the teaching time, some participants preferred to interact much longer than others, resulting in skewed data. Likewise for the testing performance: often participants either reached a successful end state or did not hit any of the sub-goals of the task in the testing phase ending often in two clusters of participants: one at a performance of 6 and one at 0. Similarly, some participants who interacted a long time with the system did not complete any step, while others could achieve good results in a limited time. Due to the data being not normally distributed and the absence of possible transformation making them normal, Bayesian statistics were conducted using the JASP software (JASP Team, 2018). Three types of test have been used: mixed ANOVA for omnibus comparisons between conditions for the first and the second interaction (between participants), independent t-tests for post-hoc comparisons between participants and paired samples t-tests for post-hoc comparisons within participants. All tests have been performed using their Bayesian counterpart, which also removed the need for doing a correction on post-hoc tests such as Bonferroni. As such, no p-value is reported, but a B factor representing how much of the variance on the metric is explained by a parameter (if $B < 1/3$ there is no impact, if $B > 3$ the impact is strong, and if $1/3 < B < 3$ the results are inconclusive; Jeffreys 1998; Dienes 2011).

For each interaction metric, two mixed ANOVA between participants were calculated to explore the impact of the conditions on the metric. The first ANOVA was applied to the first interaction (session 1,2 and 3) and compared participants in group 1 interacting with IRL and participants in group 2 interacting with SPARC. The second ANOVA was applied on the second interaction (sessions 4, 5 and 6) and compared participants in group 1 interacting with SPARC and those in group 2 interacting with IRL. If required, additional post-hoc tests were made within participants for the three sessions corresponding to each interaction to measure if successive interactions with a same system impacted the metric.

5.4.1 Interaction Data

Five objective metrics (teaching performance, testing performance, teaching time, number of inputs provided and number of failures) have been used to assess the efficiency of IRL and SPARC.

5.4.1.1 Teaching Performance

Figure 5.3 presents the maximum performance reached by participants during the teaching phase, i.e how far in the steps they brought the robot during the teaching phase. It relates to the ease of guiding the robot through the task using a method. If a method does not allow a teacher to direct the robot's behaviour, the robot will have issues reaching useful states which will lead to a poor teaching performance. On the other hand, methods allowing the teacher to steer the robot to the desired parts of the environment should achieve a high teaching performance. The teaching performance is also an upper bound for the testing performance as, due to the risk of failures or loop in the environment, the performance in the testing phase cannot (or has dramatically low probability to) be higher than the performance in the teaching phase.

In the first three sessions participants interacted with either IRL or SPARC and swapped for the remaining three sessions. The bayesian mixed ANOVA showed differences between conditions (and between participants) when interacting in the first interaction (when interacting with the first system: participants in group 1 interacting with IRL and those in group 2 with IRL) and this effect was also present in the second interaction (first interaction: $B_1 = 2881$ - second interaction $B_2 = 76.2$). According to the medians shown in Table 5.2 and the graphs in Figure 5.3, for both interaction, participants using SPARC achieved a higher teaching performance than the ones using IRL. The session number

(the repetition of additional sessions with the same system - within participants) had no impact on the teaching performance (first interaction: $B_1 = 0.089$ - second interaction $B_2 = 0.105$) which indicates that with additional interaction with a system participants did not reach a higher or lower teaching performance.

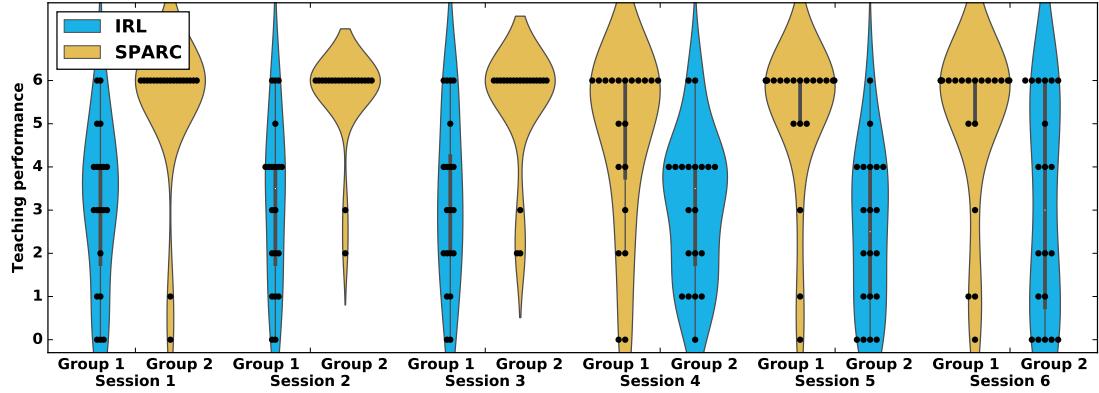


Figure 5.3: Comparison of the teaching performance for the six sessions (the left columns present the data of participants in group 1 and the right ones those in group 2). The colours are swapped between session 3 and 4 to represent swapping of conditions. A 6 in teaching performance shows that the participant reached at least one success during the teaching phase. The vertical grey lines represent minimal barplots of the data and the shaded areas the probability distribution most likely to produce these results.

Table 5.2: Median performance in the teaching phase. Noted that between session 3 and 4 participants change system.

	\tilde{X}_1	\tilde{X}_2	\tilde{X}_3		\tilde{X}_4	\tilde{X}_5	\tilde{X}_6
IRL	3.0	3.5	3.0		3.5	2.5	3.0
SPARC	6.0	6.0	6.0		6.0	6.0	6.0

This higher teaching performance for SPARC provides partial support for H1 and its prediction: 'SPARC will lead to better learning than IRL when used by non-experts'.

5.4.1.2 Testing Performance

Figure 5.4 presents the performance of the system during the testing phase, and represents how successful was the participants' teaching. The bayesian mixed ANOVA showed an effect of condition on the performance for both interactions ($B_1 = 8.8 \times 10^5$ and $B_2 = 7340$). The median performance scores in Table 5.3 show that a higher test performance was achieved when participants used SPARC compared to when they used IRL. The session number had no impact on the performance in the first interaction, but results were inconclusive for the impact of the session number during the second interaction ($B_1 = 0.084$ and $B_2 = 0.80$).

5.4. RESULTS

As shown in Table 5.3 and Figure 5.4, only a limited number of participants succeeded in teaching the robot to complete the task using IRL, this finding will be discussed in more details in section 5.6.

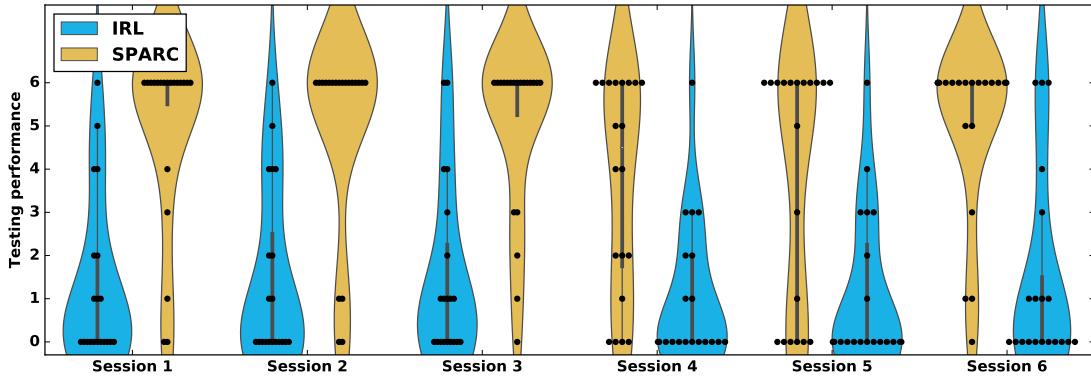


Figure 5.4: Comparison of the testing performance for the six sessions. A 6 in performance shows that the taught policy led to a success.

Table 5.3: Medians of the performance in the testing phase.

	\tilde{X}_1	\tilde{X}_2	\tilde{X}_3		\tilde{X}_4	\tilde{X}_5	\tilde{X}_6
IRL	0.0	0.0	1.0		0.0	0.0	0.0
SPARC	6.0	6.0	6.0	$\times \swarrow \searrow \times$	4.5	6.0	6.0

This higher testing performance for SPARC provides partial support for H1 and its prediction.

5.4.1.3 Teaching Time

Figure 5.5 presents the time participants spent teaching. They could stop whenever they decided or the session would stop automatically after 25 minutes. The bayesian mixed ANOVA showed the important role of condition ($B_1 = 31.4$ and $B_2 = 679$) and session number on the time spent teaching ($B_1 = 8.3 \times 10^9$ and $B_2 = 3188$). Table 5.4 and additional post-hoc comparisons between the sessions in each interaction indicated that in the first interaction, the teaching time decreased between the first and the second session and then tended to stabilise between the second and the third sessions ($B_{12} = 4.4 \times 10^5$, $B_{13} = 2.6 \times 10^6$ and $B_{23} = 0.435$). A similar pattern occurred in the second interaction ($B_{45} = 850$, $B_{46} = 382$ and $B_{56} = 0.172$) with more support for a stabilisation of teaching time between session 5 and 6.

Combined with the consistent high performance of SPARC, this decrease of teaching time indicates that participants managed to learn an efficient way to use SPARC to teach the robot a successful policy. On the other hand, this similar decrease of teaching

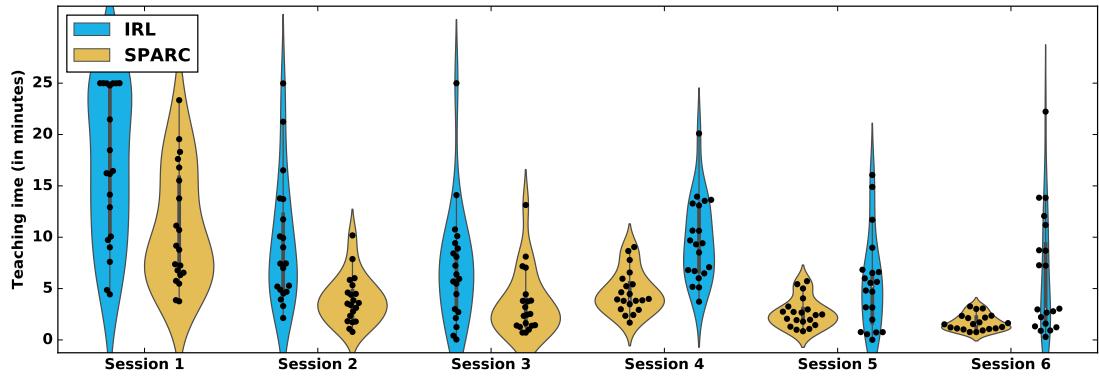


Figure 5.5: Comparison of the teaching time for the six sessions. At 25 minutes, the session stopped regardless of the participant stage in the teaching.

Table 5.4: Medians of the teaching time in each session (in minutes).

	\tilde{X}_1	\tilde{X}_2	\tilde{X}_3	\tilde{X}_4	\tilde{X}_5	\tilde{X}_6
IRL	16.34	7.43	6.16	9.36	5.18	3.0
SPARC	8.97	3.56	2.49	3.96	2.45	1.53

time and the lower performance with IRL could indicate that participants lost motivation to interact with IRL. As they did not find an efficient way to teach the robot with IRL, participants might have dedicated less efforts to try in successive session. These interpretations provide partial support to H1 and its prediction.

5.4.1.4 Number of Inputs

Figure 5.6 presents the number of inputs the participants provided while teaching. The bayesian mixed ANOVA showed that in both interactions, the condition had an impact on the number of inputs provided ($B_1 = 27.4$ and $B_2 = 34.1$). On the other hand, the session number only had a clear impact for the first interaction, the results were inconclusive for the second interaction ($B_1 = 4.1 \times 10^5$ and $B_2 = 1.5$). Table 5.5 and additional post-hoc comparisons between sessions indicated that in the first interaction, the number of inputs used decreased between the first and second sessions and then tended to stabilise between the second and the third sessions ($B_{12} = 2707$, $B_{13} = 4.7 \times 10^4$ and $B_{23} = 0.410$). Similarly, for the second interaction, a difference tended to be observed between session 4 and 5 and session 4 and 6 while the number of inputs was similar between session 5 and 6 ($B_{45} = 2.6$, $B_{46} = 2.7$ and $B_{56} = 0.17$).

Similarly to the teaching time, this reduction of inputs provided during the teaching, while maintaining a high performance for SPARC offers partial support for H1 and its prediction.

5.4. RESULTS

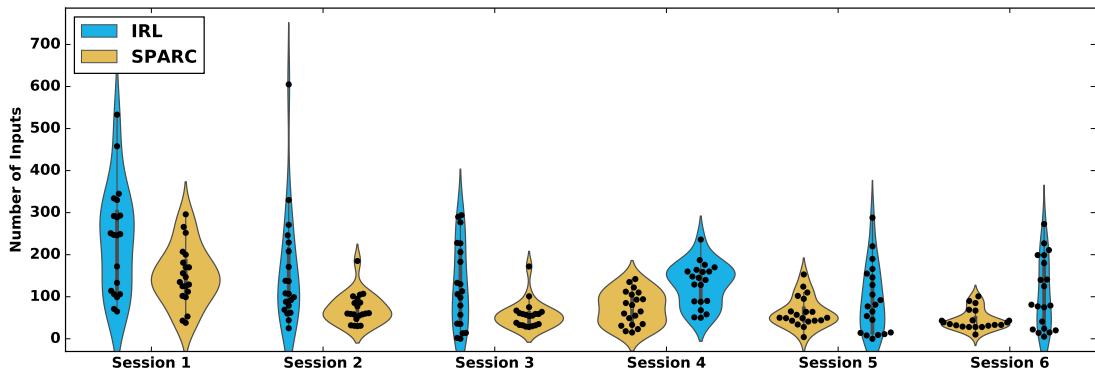


Figure 5.6: Comparison of the number of inputs provided by the participants for the six sessions.

Table 5.5: Medians of the number of inputs in the testing phase.

	\tilde{X}_1	\tilde{X}_2	\tilde{X}_3	\tilde{X}_4	\tilde{X}_5	\tilde{X}_6	
IRL	248.0	107.5	109.5		142.5	79.0	80.0
SPARC	141.0	60.0	56.0		72.5	50.0	37.0

5.4.1.5 Number of Failures

Figure 5.7 presents the number of failure states participants encountered during the teaching phase. The bayesian mixed ANOVA showed that for both interactions, both the condition ($B_1 = 6.2 \times 10^4$ and $B_2 = 2.6 \times 10^4$) and session number ($B_1 = 1.5 \times 10^4$ and $B_2 = 11$) played an important role on the number of failures. Table 5.6 and additional post-hoc comparisons between sessions indicated that in the first interaction, the number of failures decreased between the first and the second session and then stabilised between the second and the third one ($B_{12} = 619$, $B_{13} = 1.7 \times 10^3$ and $B_{23} = 0.25$). Similar results have been observed in the second interaction ($B_{45} = 3.3$, $B_{46} = 7.5$ and $B_{56} = 0.2$).

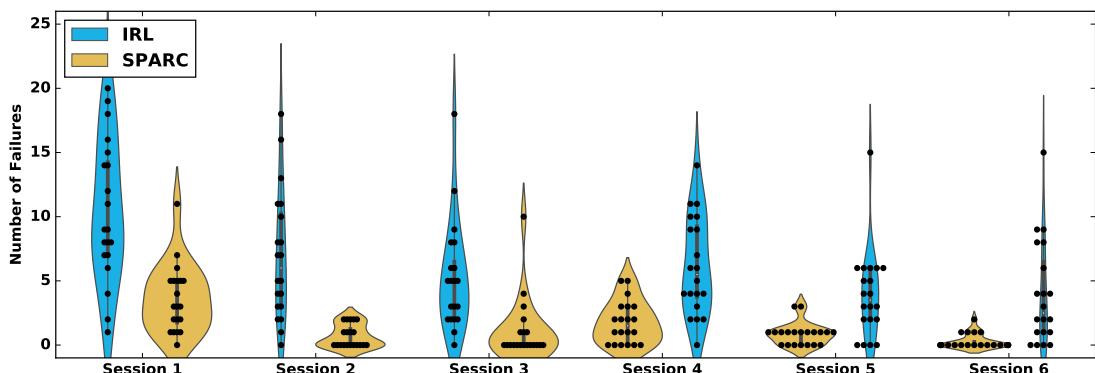


Figure 5.7: Comparison of the number of failures for the six sessions.

The fewer failures faced when using SPARC compared to IRL offers partial support to H1 and its prediction. Additionally, the low number of failures when using SPARC in

Table 5.6: Medians of the number of failures in the testing phase.

	\tilde{X}_1	\tilde{X}_2	\tilde{X}_3		\tilde{X}_4	\tilde{X}_5	\tilde{X}_6
IRL	9.0	6.0	5.0		5.5	3.5	2.5
SPARC	3.0	0.0	0.0	↗ ↘	1.5	1.0	0.0

the last sessions of both interaction (sessions 3 and 6) shows that participants became more efficient with SPARC, reaching successes without facing failures, which partially support H2: ‘SPARC can be used by expert users (knowledgeable in the interaction process) to teach a policy safely, quickly and efficiently’.

5.4.2 Questionnaire Data

The main task of the post-interaction questionnaires was to assess the workload on participants when interacting with a condition using the NASA-TLX questionnaire. Figure 5.8 presents the workload for participants for each condition for both interactions (the average of the six ratings from 0 to 20 for each category). In the first interaction, participants using IRL reported an average workload of 12.9 ($SD = 2.33$), whereas the ones using SPARC reported 8.94 ($SD = 3.01$). In the second interaction, participants interacting with IRL reported an average workload of 13.87 ($SD = 2.84$) and the ones using SPARC reported 7.44 ($SD = 3.41$). Bayesian independent t-tests show a strong effect of the condition for both interactions ($B_1 = 462$ and $B_2 = 8.1 \times 10^4$) between participants. And bayesian paired t-tests show a similar effect of the condition within participants for both orders (order 1: $B_{IRL-SPARC} = 1.7 \times 10^6$ - order 2: $B_{SPARC-IRL} = 1.1 \times 10^4$). Regardless of the comparison criteria (between or within subjects), participants reported a lower workload when interacting with SPARC than when interacting with IRL.

This lower workload when using SPARC compared to IRL offers partial support for H1 and its predictions.

5.4.3 Expert

To evaluate the best case potential offered by SPARC and IRL, an expert in Human-Robot Interaction (HRI) knowing the detail of the algorithm and the interactions (the author) interacted five times with each system. For both systems, the expert followed a strictly optimal strategy. In the case of IRL the optimal strategy consisted on providing as much guidance as possible, rewarding positively correct actions and negatively incorrect ones. For SPARC the optimal strategy consisted on providing commands for every

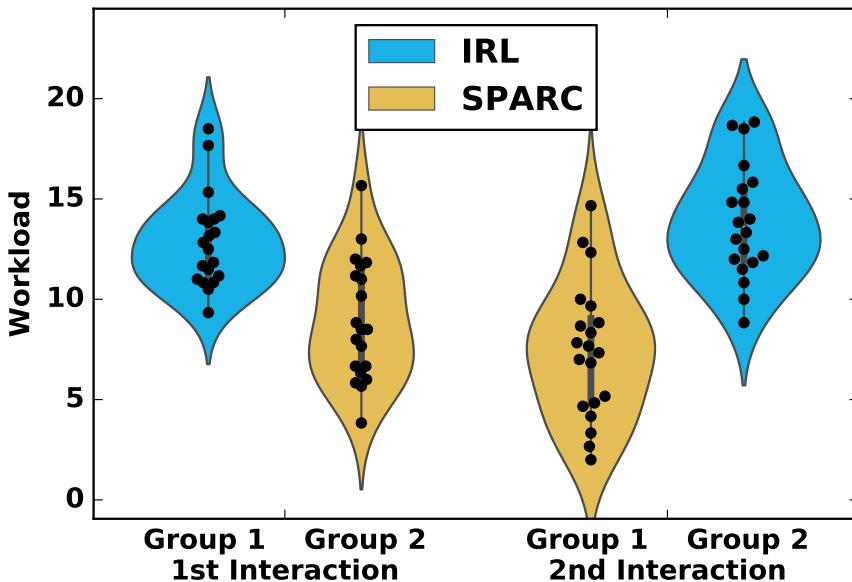


Figure 5.8: Average workload for each participants as measured by the NASA-TLX for each conditions in both interaction order.

single action to demonstrate an optimal trajectory to the robot. This shows the expected behaviours in optimal conditions, the best metrics achievable. Results of the interactions are presented in Table 5.7. In both cases, the expert successfully taught the robot (as indicated by a performance of 6 during the teaching and the test), which indicates that both systems can be used to teach a robot a policy. However as demonstrated by a bayesian independent t-test, the time required to teach the robot with IRL is higher than with SPARC ($B = 7102$). Due to the lack of variation of the number of inputs for SPARC, no Bayes factor can be computed for this metric. However, the results presented in the table clearly demonstrate that SPARC required less inputs from the expert to teach the robot a correct policy than IRL.

Table 5.7: Results of an expert interacting 5 times with each system following an optimal strategy. When the variance is 0, Bayes Factor cannot be computed.

	IRL $M(SD)$	SPARC $M(SD)$	B factor
Performance	6 (0)	6 (0)	NA
Time (minutes)	4.5 (0.67)	0.60 (0.03)	7102
Inputs	115.6 (8.4)	28 (0)	NA
Number of Failures	3.2 (0.84)	0 (0)	NA

Additionally, when using IRL, even an expert cannot prevent the robot from reaching failure states during the teaching due to the lack of control over the robot's action. In contrast, when interacting with SPARC, due to the full control and clear communication, the teacher can ensure that only desired actions are executed. So, with sufficient

knowledge of the interaction possibilities, an expert using SPARC can teach the robot to behave safely without having to explore or reach undesired states. This has real world applications in HRI, as random exploration is often impossible or undesirable when interacting with humans. SPARC offers a way for the teacher to stop the robot from executing actions with negative consequences whilst still guiding the robot toward useful parts of the environment.

Similar results to these were observed with our non-expert participants: in their last session with SPARC, both groups had a median of 0 failures and a performance of 6. This indicates that more than half of the participants successfully taught the robot the task without ever hitting a failure state after gaining understanding of SPARC in their first and second interactions with it.

The absence of failures, the lower number of inputs and the shorter time required to teach with SPARC compared to IRL when used by an expert user provide support for H2.

5.5 Validation of the Hypotheses

5.5.1 Effectiveness and Efficiency with Non-Experts

Results from the interaction data showed that despite spending a shorter time interacting with SPARC and using fewer inputs, participants reached a higher performance than with IRL and faced fewer failures during teaching. Additionally, when interacting with SPARC, the time participants took to teach the robot decreased to reach a plateau in the second and third sessions, without negatively affecting the performance. This indicates that after the first session, participants understood the interaction mechanism of SPARC and consistently managed to achieve a high performance whilst requiring less effort to teach the robot the task. On the other hand, when interacting with IRL, participants' performance remained low over the sessions, while their teaching time decreased between session 1 and 2 but not further between session 2 and 3. This decrease of effort invested in the teaching combined with the low performance might be due to a loss of motivation after session 1 where often participants did not succeed to teach the robot, reducing the desire to further interact in successive sessions. Overall, these results suggest that teaching the robot using SPARC allows the robot to achieve a higher performance than with IRL, in a shorter time, while requiring fewer inputs and making fewer errors when teaching. This conclusion is supported by subjective

measures: the workload on the teacher was lower when using SPARC than when using IRL.

For these reasons, H1 and its prediction ('SPARC will lead to better learning than IRL when used by non-experts') are supported.

5.5.2 Safety with Experts

As presented in Section 5.4.3, when interacting with SPARC, an expert can reach a success easily and safely (requiring a low number of inputs and a short time, and without facing a single failure). This effect is also observed after some training for the naive participants: most of them reached a success without encountering any failures in their last session with SPARC.

However, when interacting with IRL, even the expert applying a strictly optimal policy could not prevent the robot reaching failures states. This effect is due to the lack of control of feedback-based IML methods. As teachers only rate the actions of the agent, they cannot prevent the learners from making errors. They can only negatively reward these errors to reduce their chance of being selected in the future. While the guidance in IRL allowed to partially mitigate this effect, the presence of actions not covered by this guidance limited its efficiency during the teaching.

This difference shows support for H2 ('SPARC can be used by expert users to teach a policy safely, quickly and efficiently, achieving better results than other IML methods lacking control'). This also demonstrates how the principles presented in Chapter 3 provide the teacher with control over the robot's actions and by extend improve the teaching. Consequently, the principles underlying SPARC ensure that even in the early stages of teaching (when the robot's policy is not mature enough to correctly select actions without supervision), the robot's behaviour is already appropriate, which is not the case of most other IML methods (as demonstrated by the number of failures when teaching with IRL).

5.5.3 Control

One of the main differences between the two methods is the way in which the concept of teaching is approached. With IRL an exploratory individual learning approach is followed: the robot has the freedom to explore, whilst receiving feedback on its actions and limited guidance about what action to pursue next from a teacher. While not every

member of the population is knowledgeable about Machine Learning (ML), people are experienced with social learning and teaching (Thomaz & Breazeal, 2008). This social aspect of the teaching: with hints and guidance, partial control over the robot actions and bidirectional communication is inspired by the way humans teach and should be natural for most of the people. This similarity between how humans teach robots and other humans has also been supported by behaviours displayed by participants in the original study. Participants gave motivational rewards to the robot, just as one would do to keep a child motivated during learning, despite the absence of effect or use in classical RL (Thomaz & Breazeal, 2008).

On the other hand, SPARC promotes a more direct teaching process: the supervisor explicitly tells the robot what to do and expects it to obey and learn. The robot is not totally considered as a social agent from the supervisor's point of view, but rather as a tool having to learn a policy. This does not mean that the robot cannot be social: the supervisor can teach the robot in a non-social way how to interact socially. This approach is more task oriented, and we argue that it better fits many human-centred applications of HRI when the interaction with the teacher does not have to be social. For example, in Socially Assistive Robotics (SAR), the task (such as interaction with a child with Autism Spectrum Disorder (ASD)) is more important than the social relationship between the robot and its supervisor (a therapist for example) and as such the relevance of the social side of the interaction between the teacher and the robot is reduced.

The post-experiment questionnaire included the open question: 'which robot did you prefer interacting with and why?'. Almost all the participants (38 out of 40) replied that they preferred interacting with SPARC. Half of all the participants used vocabulary related to the control over the robot's actions ('control', 'instruction', 'command', 'what to do' or 'what I want') to justify their preferences without these words being used in the question. Furthermore, multiple participants reported being frustrated not to have total control over the robot's actions with IRL, they would have preferred being able to control each of the robot's actions.

To the question 'which interaction was more natural?', 10 participants rated IRL as being more natural, using justifications such as: 'The robot thinks for itself', 'Some confusion in the [IRL] robot was obvious making it more natural', 'More like real learning', 'Because it was hard to control the robot' or 'People learn from their mistakes faster'. But despite

these participants acknowledging that IRL is more ‘natural’, closer to human teaching, they still preferred teaching using SPARC. This suggests that when humans teach robots, they are focused on the outcome of the teaching, on the learner’s proficiency in the task. As mentioned previously, this might relate to the role of robots, they often interact in human-centred scenarios where they have to complete a task for their users. And, due to the absence of life-long learning for robots today, it is not worth investing time and energy to allow the robot to improve its learning process or explore on its own. These evaluations and comments from the participants show support for H3 (‘Teachers prefer a method in which they can have more control over the robot’s actions’).

5.6 Discussion

Despite not being originally designed for use in combination with Reinforcement Learning, SPARC achieved good results in this study. This shows that the principles presented in Chapter 3 are agnostic to the learning algorithm and promote an efficient teaching interaction. Furthermore, SPARC achieved a higher performance, in a shorter time and facing less failures than IRL, whilst requiring a lower workload from the human teacher. And finally, when used by experts (designer or trained participants), SPARC demonstrated that teaching can be safe and quick. Providing the teacher full control over the robot’s actions ensured that only desired actions will be executed. These results showed an important feature of teaching robots to interact in human environments. As robots interact in task oriented, human-centred environments, human teachers need approaches with direct control and more focused on commands rather than letting the robots explore on their own and only evaluate their actions.

5.6.1 Comparison with Original Interactive Reinforcement Learning Study

Unlike the original experiments evaluating IRL (Thomaz & Breazeal, 2008), in this study, most of the participants did not succeed in teaching the robot the full cake baking sequence using feedback and guidance (the IRL condition). In Thomaz and Breazeal’s study, the participants were knowledgeable in machine learning: when asked to rate their expertise in ML software and system (1=none, 7=very experienced), they reported an above average score ($M=3.7$, $SD=2.3$), but the population in the study presented in this chapter was drawn from a more general public having little to no knowledge of machine learning ($M=1.8$, $SD=1.13$ - on a 5-item Likert scale). This can explain why a much larger number of participants did not achieve success with IRL in this study

whereas Thomaz and Breazeal only reported 1 participant out of 13 failing the task. In our study, only 12.5% of the participants and the expert did manage to teach the robot using IRL.

As demonstrated by the teaching performance, most of the participants did not manage to reach a single success even during the teaching phase in the IRL condition. We identify the lack of control over the robot's actions as a limiting factor for the teaching. As participants did not manage to steer the robot toward correct actions, they could not reward them and teach the robot an efficient policy. Additionally, the requirement of explicit feedback made the learning task more complex. Participants often did not reward an action after a guidance, assuming that informing the robot about what it should do was enough, and would not require an additional explicit reward. In contrast, SPARC used implicit rewarding by automatically providing a positive reward to actions executed by the robot as they either have been selected by the teacher or implicitly approved. This improvement of performance with SPARC and participants' preference for this system indicates that when teaching robots, humans should be provided with control over the robots and robots should take into account teacher's implicit teaching strategies. This is consistent with Kaochar et al. (2011) who note that feedback is not well suited for teaching a policy from scratch, but better for fine tuning. For teaching the basis of a policy, they recommend using demonstrations, a method much closer to SPARC.

5.6.2 Advantages and Limitations of SPARC

In the implementation of SPARC for this study, the algorithm mostly reproduced actions selected by the teacher. One could argue that no learning algorithm was required to produce that behaviour, instead the actions could just be blindly reproduced by the robot. However, using RL with SPARC does provide advantages. By using a Markov Decision Process, a state visited multiple times in an episode would be considered as a single identical state for the learning algorithm. This means that by providing different rewards to the actions selected in that state, loops in the demonstrations can be removed for future executions of the policy. By updating the Q-table, RL provides the algorithm with a way to deal with variations in teaching and settle on a policy. Additionally, RL allows the robot to reach a success from the initial state but also to continue the policy from any state in the trajectory. And finally, due to the suggestion/correction mechanism,

the teacher can let the robot act on its own only intervening when the robot is about to execute an incorrect action. While we acknowledge that sometimes negative rewards can teach a robot more than positive ones, we argue that generally in HRI, the well-being and comfort of the user are more important than the robot's learning. Consequently, in HRI, it is more important for the robot to avoid these undesired actions rather than learn from them.

Over the 79 successful trials using SPARC, the robot learned 47 different strategies to bake a cake. This shows how SPARC, as a single control mechanism, allows for different policies to be learned depending on the person teaching the robot and their preferences. With SPARC the robot can learn the specific way its user would like it to behave and take into account their personal preferences.

However SPARC also has limitations in the current implementation. The main limits are related to the quality and interpretation of the human supervision. If the teacher allows an action to be executed by mistake (through inattention or by not responding in time), this action will be reinforced and will have to be corrected later on. This might lead to loops when successive actions are returning to a previous state (such as move left, then right). In that case, the teacher has to step in and manually guide the robot to break this cycle. Furthermore, due to the automatic execution of actions, the teacher has to be attentive at all times and ready to step in when a wrong action is suggested by the robot. This is a limitation as a lack of supervision can lead to undesired execution and reinforcement of incorrect behaviours.

5.6.3 Limitations of the Evaluation

In this study, SPARC has been applied to a scenario where a clear strategy with optimal actions was present. The interaction also took place in a virtual environment with a discrete time and a limited number of states. Real human-robot interactions are stochastic, happen in real time and often there is no clear strategy known in advance, and as such present limited similarity with this study. However, in these real HRI, human experts in the application domain know what type of actions should be executed when, and which features of the environment they used for their decision. And, as this knowledge might not be available to the robot's designers or could be complex to formalise in a set of rules a robot should follow, robots should be able to learn from

a domain user in an interactive fashion. This study presented a simple environment allowing us learn more about how humans could teach a robot using SPARC.

Additionally, the participants were also biased. By using the tool provided by the university, a majority of the participants were students from the university, and might not be fully representative of the general population. However, as robots are by essence technological tools, they will probably be mostly used by people familiar with technology or people having been trained to use such tools.

Some limitations of this study (simulated deterministic world, limited number of states, discretisation of time and absence of interaction with humans in the target application) were addressed in Chapter 6. In the study presented in that chapter, SPARC was applied to a real-world social interaction with humans, possessing all the challenges typical to these interactions: complex non-deterministic world, with continuous time, real impact of actions and importance of the social factors in the interaction.

5.7 Summary

As presented in Chapter 3, SPARC has been designed to allow naive humans to teach a policy to a robot while maintaining an appropriate behaviour. This chapter presented a study where SPARC was combined with RL to teach a simulated robot to complete a baking task. SPARC used intentions communicated by the robot, teacher's full control over the robot's behaviour and implicit rewarding to allow participants to teach the robot a policy. A study involving 40 participants compared this approach with IRL, another IML approach using communication of uncertainty, partial control and explicit rewarding to teach the robot. When interacting with SPARC, participants took less time and fewer inputs to reach more successes, whilst facing fewer failures. Participants also reported a lighter workload when using SPARC than when interacting with IRL. Results from this study demonstrated that SPARC is usable by naive participants to successfully teach a robot a policy quickly and safely.

This chapter extended SPARC and compared it to an other method from the IML field. SPARC succeeded in its goal, allowing participants to teach easily and safely a policy to a robot. Finally, these encouraging results supported us in the decision to move on to the final study of this research (presented in Chapter 6), which addressed some of the limitations of this study by exploring how a human could teach a robot to interact with children in real-world HRI.

Chapter 6

Study 3:

Application of SPARC to Tutoring

Key points:

- Design of an experiment to test SPARC in an educational application with children.
- Design and use of a new learning algorithm adapted from Nearest Neighbours to teach quickly and efficiently in an online fashion.
- Between participants study involving 75 children comparing 3 conditions: a passive robot, a supervised robot and an autonomous robot.
- Psychology PhD student teaching the supervised robot using SPARC.
- The autonomous robot learned a policy showing similarities with the supervised one.
- Children behaved similarly with the autonomous and supervised robot, achieving between 10 and 30% of improvement in game metrics compared to when interacting with the passive robot.
- Demonstration of the application of SPARC to teach a robot a policy to interact with humans *in situ*, in a social environment which is complex, non-deterministic, high dimensional and multimodal.

Parts of the work presented in this chapter have been published verbatim in Senft et al. (2017b) and Senft et al. (2018a) and an additional publication is to be submitted. The final publications are available from AAAI, EPFL:

-
- Toward Supervised Reinforcement Learning With Partial States for Social HRI. In Proceedings of the Artificial Intelligence for Human-Robot Interaction Symposium, at AAAI Fall Symposium Series¹.
 - Robots in the Classroom: Learning to Be a Good Tutor. In 4th Workshop on Robots for Learning (R4L) at HRI².

¹<https://aaai.org/ocs/index.php/FSS/FSS17/paper/view/16011>

²https://r4l.epfl.ch/files/content/sites/r4l/files/HRI2018/proceedings_2018/paper4.pdf

6.1 Motivation

Chapters 4 and 5 tested Supervised Progressively Autonomous Robot Competencies (SPARC) in interactions between robots or in a virtual world but not for Human-Robot Interaction (HRI) as it was intended to be used. As such, this chapter addresses the thesis of this research and evaluates whether SPARC can be used to efficiently teach a robot an interactive behaviour for real human-robot interactions. HRI in the wild typically occurs in constrained but underspecified environments where social behaviours play an important role. Teaching a robot in such an environment is a challenge as the state and action spaces are high-dimensional, the environment is not deterministic and the interaction takes place in continuous time and is multimodal. Additionally, the social aspect of the interaction is fundamental as it impacts the flow and the success of the interaction and makes HRI a high-stakes environment.

This study took place in the context of robot tutors for children in education, more precisely in the context of teaching about food chains. Child tutoring has been selected as this framework is widely used in HRI, and provides opportunities for a rich and complex interaction between a child and a robot (Leyzberg et al., 2012; Kennedy et al., 2015a; Belpaeme et al., 2018). This scenario, the code used to control the robot and the educational software are based on Lemaignan et al. (2018) but have been adapted to provide a new teaching task (teaching game and protocol), knowledge test, robot controller, learning algorithm and interface with the teacher supporting SPARC.

6.2 Scope of the Study

The main goal of this study was to directly explore the thesis proposed in this research work: “Through receiving supervision from a human teacher, a robot can progressively and efficiently learn a policy for interacting with people, during the learning the robot will at all times display correct behaviour”. This thesis can be divided into two parts: “by receiving supervision from a human teacher, a robot can learn while only displaying an appropriate behaviour” and “after learning, such a robot would be able to interact autonomously and efficiently with humans”. To address these two statements, a study comparing three conditions was designed (passive robot vs supervised robot vs autonomous robot). The study was composed of two main elements: four rounds of an educational game children played with the robot where they could gain knowledge about food chains, and three tests outside of the game to evaluate the knowledge children

gained while playing. Depending on the condition, the robot could provide feedback, hints and supporting messages to the child participating during the game.

The objective of the study was to explore whether the robot could be taught to provide an efficient tutoring support to the children during this game, and examining how the resulting autonomous behaviour would compare to no behaviour (passive) and to human-controlled behaviour. As such, three conditions were required: a control condition (with a passive robot), a supervised condition (where the robot was controlled and taught by a human) and an autonomous condition (to evaluate the learned behaviour). In both the supervised and the autonomous conditions, the robot was actively participating during the game and provided support to the child.

As such, there were four experimental hypotheses:

H1 In the supervised condition, the teacher will be able to ensure an appropriate robot behaviour whilst teaching.

H2 The autonomous robot will be able to interact socially and efficiently during the game and maintain the child's engagement during the learning task.

H3 An active robot (supervised or autonomous) supports child learning: the learning gain in the passive condition will be inferior to the learning gain in the autonomous condition, which will be inferior to the learning gain in the supervised condition.

H4 Using SPARC, the supervisor's workload decreases over time: the number of corrected actions and the number of actions selected by the teacher decrease with practice, while the number of accepted proposed actions increases.

H3 is motivated by the idea that the humans possess knowledge which should help the child to learn more from the game. By learning this knowledge, the autonomous robot should be able to partially replicate this effect, but without being able to match it due to the limits of the algorithm and teaching time.

6.3 Apparatus

Similarly to the study presented in Chapter 4, this study is based on the Sandtray paradigm (Baxter et al., 2012): a child interacts with a robot via a large touchscreen located between them (hereafter 'Sandtray' or touchscreen). By interacting with the touchscreen and the robot, the child is expected to gain knowledge or improve some

skills. Additionally, a teacher can control and teach the robot in the ‘supervised’ condition using a tablet (cf. Figure 6.1). This type of triadic interaction is typical of the interactions we considered when framing this research (cf. Figure 1.1): a human knows how the robot should behave and can supervise it and teach it how to interact with another human *in situ* by using SPARC.



Figure 6.1: Setup used in the study: a child interacts with the robot tutor, with a large touchscreen sitting between them displaying the learning activity; a human teacher provides supervision to the robot through a tablet and monitors the robot learning.

6.3.1 Food Chain Game

The main learning activity used to teach the child about food chains is a game composed of ten interactive animals (a mouse, a grasshopper, an eagle, a frog, a snake, a fly, a wolf, a small bird, a butterfly and a dragonfly) which can be moved by the child and three types of plants (4 wheat plants, 4 apples and 3 flowers, for a total of 11 plants) that are static. Animals have energy which decreases over time and they have to eat to increase their energy levels. When an animal’s energy reaches 0, the animal dies and is removed from the game. Figure 6.2 presents an example of the game screen in the middle of a round. Animals only move if the child or the robot moves them and can eat or be eaten only when the child puts them in contact with another animal or a plant. If an animal is put in contact with something it does not eat or is not eaten by, it simply makes a noise of disgust or fright. The child is instructed to keep the animals

alive as long as possible, and consequently has to feed the animals by moving them to their food. By feeding the animals, the child can learn the animals' diet. The game stops when three or more animals run out of energy.



Figure 6.2: Example of the game. Animals have energy in green and have to eat plants or other animals to survive. The child's task is to keep animals alive as long as possible.

6.3.2 Test

The test is the second activity of the study and is used to evaluate children's knowledge about the food web used in the game. This test is an open graph where children have to connect animals to their food. Figure 6.3 shows two examples of the test, with or without all the correct connections. Children are instructed to connect as many animals to their food as possible. During the first test (the pre-test), the teacher demonstrates how to connect animals by drawing an arrow from the frog to the fly, and then removing the arrow by pressing the X button in the middle of the arrow. When children think they are done, they can press the 'Continue' button, showing a screen confirming that they want to quit the test or allowing them to go back and keep connecting animals. Additionally, the robot informs the child if there are animals not connected to a food or that animals can eat many types of food if no more than one animal is connected to multiple items. In total, there are 25 correct connections and 95 possible incorrect ones.

6.3.3 Robot Behaviour

In each condition, the robot verbally leads the child through the interaction by: asking them to fill in the demographic questions, explaining the tutorial and guiding them

6.3. APPARATUS

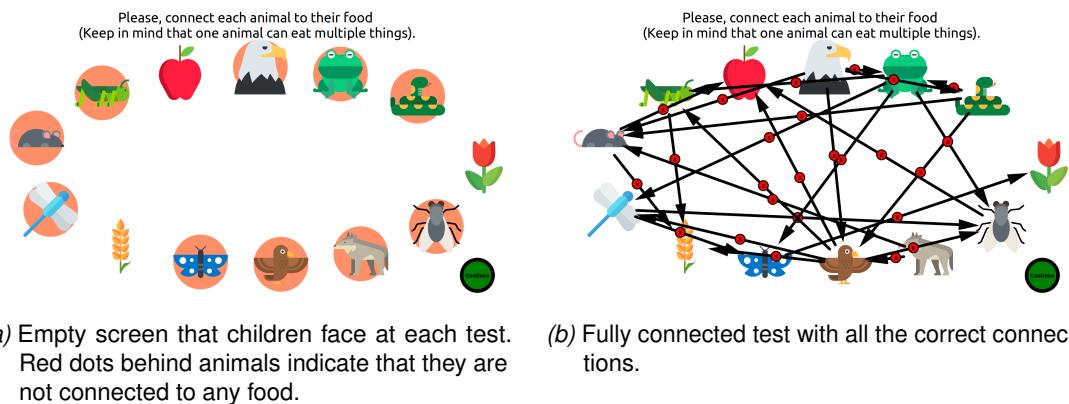


Figure 6.3: Test screen to evaluate children's knowledge, empty starting screen (a) and fully connected and correct test (b).

through the test steps. During the game, depending on the condition, the robot can execute actions to provide hints and support to the child. The robot has access to five types of actions:

- Movements: moving any animal *to, close to or away from* any items (animal or plant) - the robot points to an animal, the image of a robot hand appears on the screen and moves the animal's image. This movement is synchronised with a physical gesture from the robot and a verbal description of the action (e.g. "The eagle needs help getting close to the mouse").
- Drawing attention: the robot points to an item and gives a verbal reminder to the child (e.g. "Don't forget the frog").
- Reminding rules: the robot says one of 5 sentences describing the game's rules (e.g. "Move the animals to feed them" or "Feed animals with a low energy").
- Congratulation: the robot provides congratulations (e.g. "Well done").
- Encouragement: the robot provides encouragement (e.g. "You can do it").

Considering all the possible combinations of actions and items, the total number of actions equals 655. Additionally, to prevent the robot's behaviour from being repetitive, each action has multiple possible utterances, and a random one, not used recently, is spoken by the robot when an action is executed. When the robot is not acting, it simply sways slightly to simulate a 'breathing' motion and moves its head to follow the child's face.

This set of actions was designed to be multimodal: involving verbal utterances, pointing gestures and movements on the game and to cover different ranges of tutoring

behaviours. Some actions, such as encouragement and congratulations, are mostly social and do not provide much information on the game, others are only hints such as drawing attention, and finally the last ones: motions and reminding rules provide game content. In general, the timing of these actions is related to game events or social norms (such as turn taking). In summary, this action set aims to allow the robot to express different types of tutoring behaviour: from providing motivation and general information about the game's goal to giving hints regarding which animals the child should focus on or information about what an animal eats.

The goal of the interaction is to have the child, not the robot, play the game. Therefore, the robot can only move the animals to locations on the game, and these motions do not trigger eating events. Only the child can feed the animals by putting them in contact with a food source, hence, all the events and real changes in the game are created by the child.

6.3.4 Control Architecture

The code used to create the game and control the robot is an adaptation of the Freeplay Sandbox (Lemaignan et al., 2018). Original sources are available online³, as is the code used for this study⁴.

The control architecture is also adapted from Lemaignan et al. (2018) and a simplified schematic is presented in Figure 6.4. All nodes are communicating through the Robot Operating System (ROS) (Quigley et al., 2009) and the communication with the robot is done through NAOqi, Softbank's operating system for the Nao robot. The tool rosbag⁵ is used to collect data throughout the interaction and the analysis of the child's head orientation is performed using gavr (Lemaignan et al., 2016).

The control architecture can be divided into 7 main parts.

1) The Game Interface is responsible for displaying the game and the test on the touchscreen and generating game related data used to control the robot. The interface is coded in QML, a markup language for creating user interface-centric applications, and a ROS-QML bridge adapted from the one in Lemaignan et al. (2018) provides a

³<https://github.com/freeplay-sandbox>

⁴<https://github.com/emmanuel-senft/freeplay-sandbox-qt/tree/food-chain>
<https://github.com/emmanuel-senft/freeplay-sandbox-ros-sparc/tree/task>

⁵<http://wiki.ros.org/rosbag>

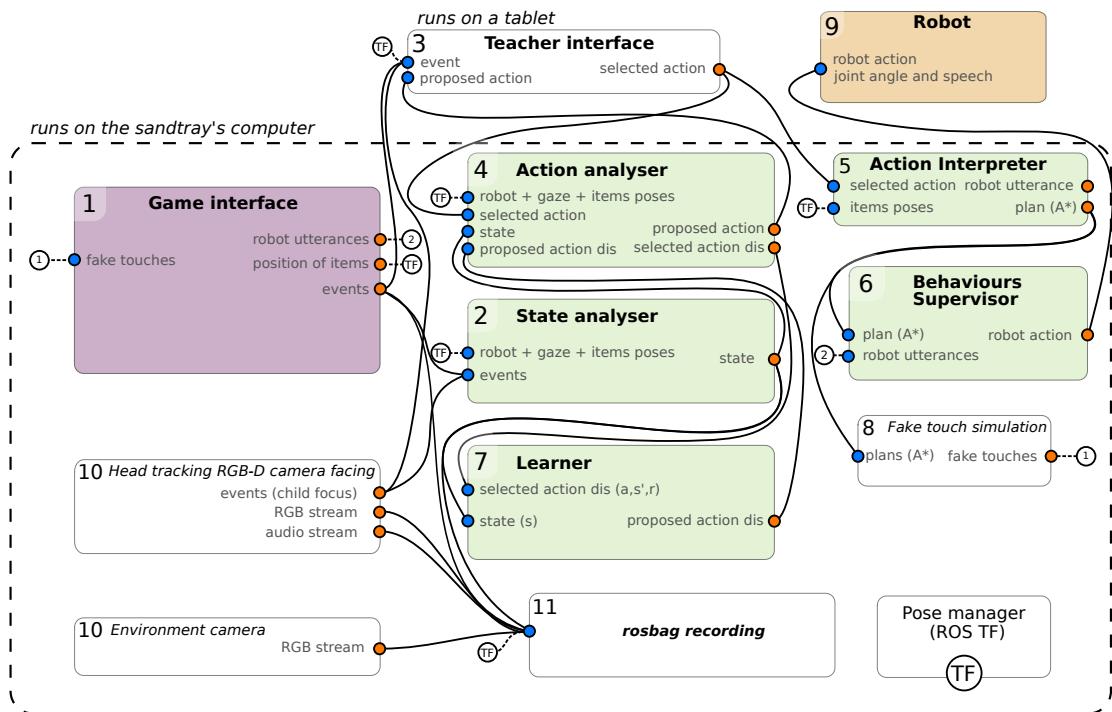


Figure 6.4: Simplified schematics of the architecture used to control the robot. In addition, all the topics related to the actions are recorded using rosbag. (Figure adapted from Lemaignan et al. 2018).

way to send and receive message from ROS, thus connecting the interface with the rest of the control architecture.

2) The State Analyser collects data from the interface and camera and from these continuous data and events generates a 210-dimensional state used for the learning (cf. Section 6.3.5.1). The state is generated at 2Hz, the frequency used to update the animals life due to the hunger.

3) The Teacher Interface is the tool used by the teacher to control and teach the robot. Similar to the game interface, it is coded in QML but runs on an additional tablet held by the teacher. The teacher can use this interface to send actions to the robot to execute (such as movements or verbal feedback) and react to propositions from the algorithm. Two clarifications should be made. First, on the interface, the teacher does not select the discrete actions for motions described in Section 6.3.3, they create continuous actions, such as moving an animal from position A to B, and select the items on the screen required for interpreting this action. The exact discrete action among the 655 presented earlier is inferred by the action analyser (cf. Figure 6.4 - node 4) and used by the learner (cf. Figure 6.4 - node 7). Additionally, by highlighting items on the screen, the teacher can inform the algorithm about the features relevant to the

selection of this action in that state. Each time the teacher selects an action or reacts to a proposition, it sends a message to the action analyser containing the continuous action (motion of an animal from A to B, or a string informing the type of action for the verbal feedback), the features relevant to the selection (as a list of strings of highlighted items) and a reward related to the acceptance or refusal of the action. Additional details can be found in Section 6.3.7.

In the autonomous condition, the teacher interface is replaced by a node running on the Sandtray which automatically accepts the actions proposed by the learner after a short delay.

4) The Action Analyser transforms the continuous actions from the teacher (such as moving an animal from point A to B) into discrete actions (moving an animal to, close to or away from an item). By using the features selected by the teacher on the interface, and the movements if applicable, this node infers a discrete action (a number from 0 to 654) and a state mask (informing the algorithm of the relevant dimensions of the state space) that are sent with the reward to the algorithm for learning (cf. Section 6.3.6). In case of ambiguity (for example if no target is selected when moving an animal), the action analyser will assume that the target was the closest item to the arriving position and will automatically use this feature to infer the action and the state mask. Similarly, this node also transforms discrete actions proposed by the algorithm into continuous actions to submit to the teacher through the interface: for example finding a location (set of coordinates on the screen) to which an animal should be moved to be close to another item.

5) The Action Interpreter and 6) the Behaviour Supervisor take the actions selected by the teacher and transform them into motor commands, simulated touches on the screen (to move the image as if the robot was physically touching the screen) and verbal utterances for the robot.

7) The Learner is the algorithm responsible for learning a behaviour and selecting actions to propose to the teacher. It takes as input the state space at each time step and actions selected by the teacher (action number, state mask and reward) and stores corresponding instances in memory. Each time the learner receives a state, it compares this state to its instances in memory and proposes an action if appropriate (cf. Section 6.3.6).

6.3.5 Environment Model

To learn and act autonomously, the robot has access to a representation of the state of the interaction and a set of actions it can perform. The algorithm aims at finding the ideal action for each possible state based on the teacher's commands.

6.3.5.1 State

Table 6.1 presents the different dimensions composing the state the robot has access to. The state analyser is responsible for updating and broadcasting this state at 2Hz.

Table 6.1: Definition of each category of the state space.

Group	Name	#	Description
Game State	Distance Between Items	155	Normalised distance between the animals and each other animal and the plants
	Items' Energy	21	Energy of the 21 items (10 animals and 11 plants)
	Progress in the Game	1	0.25 for first game to 1 for the last game
Temporality	Robot Touches	10	Time since the robot touched each animal
	Child Touches	10	Time since the child touched each animal
	Task-Specific Events	3	Time since last feeding, failed interaction and death of any animal
	Robot Actions	5	Time since last robot action of each type
	Generic Last Actions	2	Time since last child touch and last robot action of any type
Child state	Focus	3	Time of last head facing the robot, the screen and outside

All the state values in Table 6.1 are bounded by [0,1]. Distances between items are normalised with the maximum possible distance (the diagonal of the screen), and as there are 10 animals and 11 plants this results in a total of 155 possible distances between animals and other items. To include a representation of the temporal aspects of the interaction, some dimensions of the state represent the time since events and two methods have been used to transform the time since an event to a value bounded between 0 and 1: the accumulation and the decay.

The accumulation corresponds to effects that can be true or not and where the duration since a change matters (for instance how long the child has been touching an animal, or when was the last time the child looked at the robot). The accumulation aims at

representing how long this effect has been true or false. When the effect becomes true for accumulation, the state value is set to 0.5 and then increases at each step. When the condition becomes false, the value is set to 0.5 and then decreases at each step (cf. Algorithm 4 and Figure 6.5). On the other hand, the decay refers to temporally discrete events, so only the time since the event is relevant. When the event happens the corresponding state value is set to 1 and then exponentially decreases by being multiplied by $e^{-\frac{1}{10}}$. That way events will have a ‘half-life’ of 7 steps, which corresponds to 3.5 seconds, this aims to allow both short term reactions to an event (less than three seconds) and inform when an event did not happen in the last 10 or 20 seconds.

Algorithm 4: Formula to compute accumulation effects.

```

while Running do
    if effect then
        if value < 0.5 then
            | value = 0.5
        else
            | value =  $1 - (1 - value) \cdot e^{-1/10}$ 
    else
        if value > 0.5 then
            | value = 0.5
        else
            | value = value  $\cdot e^{-1/10}$ 
```

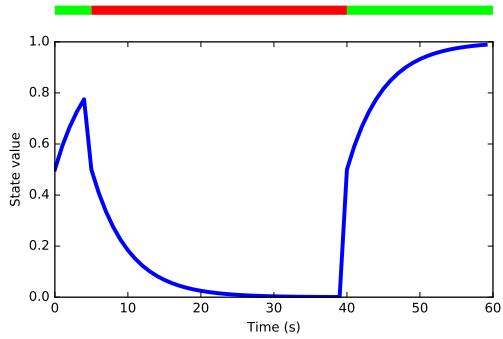


Figure 6.5: Example of computation of accumulation. When the effect is true (green), the value increases, when the effect is false (red), the value decreases and the state is set to 0.5 when the effect’s value changes.

Parts of the states are hardcoded, such as the task-specific events, generic last actions, focus and progress in the game. However, all the state dimensions related to the items (distance, energy and time since touches) are constructed on the fly. At start up, the state analyser node (responsible for computing the state, cf. Figure 6.4), receives a list of the interactive items and one of the static ones and creates the appropriate state with the corresponding dimensions. During the game, the state values are updated using events related to specific items (through their name) and hardcoded relations between task-specific events and state dimensions.

6.3.5.2 Actions

Table 6.2 present the list of actions used in the game activity.

Table 6.2: Definition of each category of the action space.

Type	Name	#	Description
Game information	Move close	210	Action moving any animal close to any item
	Move to	210	Action moving any animal to any item
	Move away	210	Action moving any animal away from any item
Social Feedback	Remind rules	1	Verbal utterance
	Congratulation	1	Verbal utterance
	Encouragement	1	Verbal utterance
Hint	Attention	21	Drawing attention to any item
Meta-level	Wait	1	Doing nothing

The total action space available is 655 actions. Each *Move* action is composed of 200 possible actions the robot can execute, as we have a total of 10 animals which can be moved in relation to each other animal (9) and each plant (11), which results in 200 actions for moving close to, 200 for moving to and 200 for moving away. It should be noted that technically 10 additional actions are available to the action analyser for each type of move (resulting in 210 actions per type of motion shown in Table 6.2), but cannot be selected by the teacher (moving animals to/close to/away from themselves). However, as the learning algorithm is instance-based and only uses the demonstration from the teacher, this does not impact the learning.

6.3.5.3 Generalisability of the State and Action Spaces

The state and action spaces have been designed to be adaptable for many tasks involving movable items on a touchscreen and not constrained to this game.

The robot has access to a wide range of actions and the action space includes little semantic specific to this game. For example, many of the actions are available even if they are not useful to this specific game (such as moving two unrelated animals close together), as they could be important if the rules were different and we don't want to limit the diversity of actions available to the robot. This allows this definition of actions to be repurposed (and by extension the learning mechanism) even if the game changes. For this specific game, some actions are important and encompass the technical knowledge the tutor needs to have (such as which animal should be moved close to which items), while others are related to the social aspect of tutoring, such as providing encouraging feedback or congratulations. Some other actions should simply be avoided as they

could create confusion for the child (such as moving the eagle close to the flower, as the eagle cannot eat it).

Similarly, the state dimensions have been selected to be generic to many teaching tasks involving interactive items: each item has a value assigned to it (herein energy, but this could be changed), and some items can be moved (here animals) while others are static (here plants). With the rest of the state, these dimensions represent both elements that define the state of the game itself, and events around which a social policy could be constructed. For example, the task-specific events ‘last feeding’ and ‘last failed interaction’ could be interpreted as a successful and a failed child action, respectively, and each trigger different social responses from the robot. However, while having a semantic interpretation for humans, the state does not represent them as positive or negative events, but only as two dimensions of the state that could be used to select actions. For the state definition, all the dimensions are considered in the same way, just as numbers in the space.

Finally, both for the state and the action spaces, each instance of a plant is considered as a static item regardless of its type. For example, each one of the four instances of wheat is considered as a static item, in the same way as flowers or apples. No grouping is made according to the plant type when considering which action has been made or should be proposed and no relation between values of states for plants of the same type is hardcoded. It could have been possible to group the instances by categories, but it would require some add-hoc coding, limiting the adaptability of the approach.

In summary, without having access to any semantics about the actions, state dimensions or rules of the game, the robot needs to learn a policy relevant to the task which includes important actions and their timing. By using this generic state definition and agnostic way of considering state values, different types of policy could be taught to the robot: for example a supporting policy (providing support and help to the child), an adversarial one (trying to prevent the child from feeding animals) or a cheeky one (trying to trick the child in a playful way, moving animals just after the child or sometimes providing misleading information). Additionally, it should be possible for this implementation to be repurposed for another teaching task with limited effort. The task-specific events and utterances for the actions would have to be changed, but to adapt the state or action definition to another set of items, only the new names would have to be communicated at start-up,

without any change in the code. If the robot manages to learn an appropriate behaviour for this specific interaction, we would have demonstrated that SPARC allows a human to teach a robot an efficient and precise policy from a generic state and action space *in situ*. This approach could be applied to another task, defining a different appropriate robot behaviour with limited changes to the code.

6.3.6 Learning Algorithm

The learning algorithm aims to reproduce the teacher's policy by mapping an action (or no action) to each possible state. The state used in this study represents the situation of the game in a 210 dimensional vector, with continuous values from 0 to 1 and the action space is composed of 655 discrete actions. In summary, the algorithm's task is to map one action from the 655 to each possible combination on the 210-dimension state. Learning in such a high-dimensional space with limited datapoints would be difficult to achieve without human feedback or initial knowledge.

The algorithm used for the learning is an adaptation of the one presented in Senft et al. (2017b). It is an instance-based algorithm similar to the Nearest Neighbours algorithm (Cover & Hart, 1967). However, two differences are notable compared to the original algorithm:

Slicing of the state space. Instead of being defined on the full state space, instances are only defined on a sliced version of the state. The intuition is that states needed to cover complex policies require large numbers of dimensions, however for each single action, large parts of the state are irrelevant. For example, if a robot needs to pick-up a cup, the colour of the cup does not impact the optimal motion. In contrast, the colour matters if the robot has to answer the question: "which cup is on the left? The blue one or the red one?". Consequently, the colour of the cup should be part of the state space, but should not be considered when selecting some actions. To implement this dimension reduction, when selecting an action with SPARC, the teacher has the opportunity to specify the features of the environment relevant to the selected action. Then these features *activate* a limited number of the state dimensions related to the selection of this action in that state (through a mask informing which dimensions should be taken into account). Later, when selecting an action, the algorithm seeks for the instance in memory closest to the current state. In this evaluation, the similarity between the current state and the stored instances is only computed on the activated dimensions

of the instances stored in memory. With this way of providing dimension reduction, the algorithm can have access to a large state, potentially covering different complex policies, but can still learn fast as if it were in a smaller state. Thus, by ignoring irrelevant dimensions, the algorithm can learn quickly. This opportunity to slice the state to reduce the number of dimensions and learn quickly in large environments is only possible with the human's supervision. The features of the environment used by the human to decide which action to select are a key part of the human's knowledge, and associating each action to the related features allows the algorithm to encode the human expertise more precisely, thus leading to a more potent learning.

Inclusion of rewards. The second difference between this algorithm and the original one (Cover & Hart, 1967) is that each instance saved has a reward assigned to it. This reward can be provided by the environment or the teacher and is used to inform the correctness of the action in that state. As argued by Knox & Stone (2009), because the human takes into account the future impacts of an action, when learning from human supervision, the algorithm can select actions myopically, without having to iteratively update a value associated to the state or the state-action pair. When selecting an action, the algorithm looks through all the actions it has been using and for each action selects the instance most similar to the current state (by taking into account only the activated dimensions of the stored instance). It then computes the expected reward as a multiplication of the similarity by the reward. Then the algorithm selects the action with the highest expected reward and, if the value is higher than an adaptive threshold, proposes it to the teacher in the supervised condition (cf. Algorithm 5).

For this implementation, when the teacher highlights items relevant to the desired action, the corresponding dimensions of the state space are activated (see Table 6.3) and the current state values on these activated dimensions are stored in memory as an instance. These connections have been hardcoded in the form of rules defining how features correspond to dimensions. When transferred to the algorithm for storing, the instance is composed of three objects: the action number (integer between 0 and 654), a mask of the state's dimension (210 boolean values) with a value of 1 for the activated dimensions and 0 for the others (this mask is used to create the sliced state corresponding to the instance), and finally, the value of the reward. The state mask is created by the action analyser by activating the dimensions corresponding to the features selected by the

Algorithm 5: Algorithm for selecting an action based on the previous instances tuples (action, sliced state, reward) and the current state. Sliced states (s') are defined on a subset of the state space, with N' the ensemble of the n' indexes of the activated dimensions of s' .

```
inputs : s: current state
        C: collection of instances  $c = (a, s', r)$ 
        A: ensemble of actions present in C
outputs :selected action  $\pi(s)$ 
foreach  $a \in A$  do
    foreach  $p = (s', r) \in C_a$  do
        compute similarity  $\Delta$  between  $s$  and  $s'$ :  $\Delta(p) = 1 - \frac{\sum_{i \in N'} (s'(i) - s(i))^2}{n'}$ 
        find closest pair  $\hat{p}$ :
         $\hat{p} = \arg \max_p \Delta(p)$ 
        compute expected reward  $\hat{r}(a)$  for taking  $a$  in state  $s$ :
         $\hat{r}(a) = \Delta(\hat{p}) \cdot r(\hat{p})$ 
        with  $r(p)$  the reward  $r$  of the pair  $p = (s', r)$ 
    Select the action with the maximum expected reward:  $\pi(s) = \arg \max_a \hat{r}(a)$ 
    if  $\hat{r}(\pi(s)) > \text{threshold}$  then
        Propose  $\pi(s)$  to supervisor
```

teacher. By default, all the task-specific events (death, feeding, failed interaction), the times since the last child and robot actions as well as the time since the prospective action was executed are activated. Later, when comparing the instance in memory to the current state to select an action, only the dimensions with a mask value of 1 are taken into account to compute the similarity.

This approach, specifying related features for a demonstration has also been used by Basu et al. (2018), however while Basu et al. require the teacher to specify the features *a posteriori*, we think that this features specification should be provided at the same time as the demonstration, allowing it to be used for online learning.

To give enough flexibility in the timing of the suggested actions (to react in time to game events) and enough time to compute the distance between the current state and each instance in memory, the algorithm runs at 2Hz (the rate of the state update). So, unlike most of the discrete cases of action selection (such as a Reinforcement Learning (RL) agent learning in a virtual world), in most time steps no action should be executed. To handle this difference of timescale, a waiting action has been added (accessed by the teacher through the ‘Skip’ button) and an adaptive threshold limits the propositions to actions with a high expected reward. As shown in Algorithm 6, when receiving an action with a positive reward (i.e. when the teacher selects an action), if the threshold is higher than the previous maximum similarity between this action’s instances and the current

Table 6.3: Example of dimension activation. By selecting features, the teacher can inform which dimensions of the state are relevant. By default, task-specific events, generic last actions and the time since the last execution of the selected action are activated.

Action	Teacher-selected features	Dimensions activated (#)	Total
Move eagle close to mouse	eagle and mouse	distance between eagle and mouse eagle's energy mouse's energy time since robot touched eagle time since robot touched mouse time since child touched eagle time since child touched mouse task-specific events (3) time since last 'move' action generic last action (2)	13
Draw attention to frog	frog	frog's energy time since robot touched frog time since child touched frog task-specific events (3) time since last 'drawing attention' action generic last actions (2)	9

state, the threshold would be decreased by a tenth of the difference as the action should have been selected; and this effect is inverted for negative rewards (the threshold would be increased by a fifth of the difference as the action should not have been selected). These increase and decrease factors have been selected from simulations which led to good results. This adaptive threshold aims to adapt the rate of action proposition to the desires of the teacher. A last mechanism filters propositions from the algorithm to prevent them from being transferred to the teacher when an action has already been proposed, when the teacher is selecting an action or if the robot is currently acting. This filter also automatically rewards negatively impossible actions (such as moving dead animals).

6.3.7 Teacher Interface

In this study, the communication between the teacher and the robot occurs through the teacher interface, a Graphical User Interface (GUI) running on a tablet and allowing the teacher and the robot to communicate (as seen in Figure 6.6). This GUI displays the state of the game as the child sees it, with additional buttons for bidirectional

Algorithm 6: Algorithm for adding one instance to the instance collection.

inputs :

s : current state

C : collection of instances $c = (a, s', r)$

A : ensemble of actions present in C

$c_n = (a_n, s'_n, r_n)$: new instance to add to C

t : threshold used for action selection

if a_n in A **then**

similarity with closest instance for this action: $\Delta = \max_{s' \in C_{a_n}} (1 - \frac{\sum_{i \in N'} (s'(i) - s(i))^2}{n'})$

if $r_n > 0$ and $\Delta < t$ **then**

action should have been selected, threshold is decreased

$$t = t - \frac{t - \Delta}{10}$$

if $r_n < 0$ and $\Delta > t$ **then**

action should not have been selected, threshold is increased

$$t = t + \frac{\Delta - t}{5}$$

add c_n to C

communication with the robot (see Figure 6.6). This interface plays two roles: allowing the teacher to select actions for the robot to execute, and to respond to and evaluate the robot's suggestions, monitoring and guiding the robot's learning. The GUI aims to provide these two functionalities in parallel.

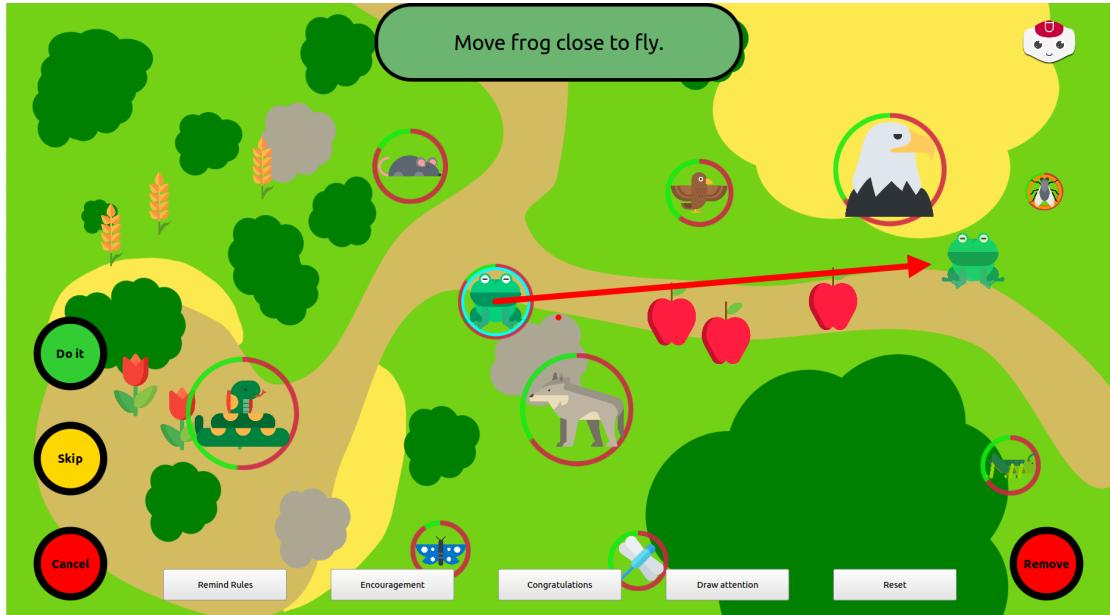


Figure 6.6: GUI used by the teacher to control the robot and respond to its suggestions.

The game is in the same state as in Figure 6.2, and the robot proposes to move the frog close to the fly (text bubble, arrow, moving the *ghost* of the frog - a partially transparent image of animal used to determine the position of a move action - and highlight the frog and the fly).

Action selection. The teacher can use the buttons and movable animals on the tablet to have the robot execute any possible action in a fashion similar to Wizard-of-Oz (WoZ).

For example, to move animals close to another item the teacher can press on an animal on the tablet, which creates a draggable ‘ghost’ of the animal the teacher can move to the desired position. On release of the ghost, a request is sent to the robot to make a moving pointing gesture synchronised with the animal’s image being moved by the image of a robotic hand on the touchscreen. As mentioned in Section 6.3.4, the action analyser determines which action has been selected by the teacher by evaluating how the distances between animals change with this motion. Other actions such as ‘Remind rules’, ‘Encouragements’ and ‘Congratulations’ can be selected by simply pressing the associated button. Additionally, the teacher can highlight the features they used to select the action to speed up the teaching process and clarify otherwise ambiguous actions. This feature highlighting is done by selecting the items relevant to action or by pressing on the icon in top-right corner (this icon indicates where the child is currently facing: the robot, the touchscreen or ‘away’ - for instance this could be used to have actions executed when the child is looking away, thus helping them to refocus on the game). The teacher can also use this feature highlighting to select an animal and press the ‘Draw attention’ button to have the robot point at the selected animal and remind the child to interact with it. Our GUI design is intended to be intuitive and simple to use, giving the teacher access to more than 600 actions without requiring as many buttons.

Evaluation of propositions. Secondly, the GUI allows the teacher to respond to the robot’s propositions. An action is proposed in a bubble at the top of the GUI describing the action, the corresponding items are highlighted and if the action is a motion, an arrow shows the proposed motion (Figure 6.6). The teacher can respond to the proposed action by pressing the ‘Do it’, ‘Skip’, ‘Cancel’ or ‘Remove’ buttons or let the action be automatically executed after 2 seconds during which the bubble will become more translucent to represent the passive acceptance of the action. The ‘Do it’ button executes the action straight-away, the ‘Skip’ button sends a ‘wait’ action with a reward of 1 (indicating the robot should not do any action at this moment but should keep suggesting actions later), the ‘Cancel’ sends the proposed action with a reward of -1 and does not execute it (indicating that this action is not appropriate in this situation but other actions might be), and finally, the ‘Remove’ button looks for the closest previous instance of the action in memory and removes it, preventing this instance from being used later. All the actions executed by the robot (through manual selection or automatic execution) are assigned a reward of 1.

These two mechanisms (selection of actions and reaction to propositions) run in parallel. The teacher is expected to both evaluate actions from the robot and select correct actions to execute and the teacher interface aims to make this feasible and as easy as possible.

6.4 Methodology

6.4.1 Participants

Children from five classrooms across two different primary schools in Plymouth were recruited to take part in the study. As both schools have an identical OFSTED evaluation (indicating that they provide similar educational environments), all the children were combined into a single pool of participants. Full permission to take part in the study and be recorded on video was acquired for all the participants via informed consent from parents. In total, 119 children participated in the study: 75 children were included in the final analysis. 14 participants took part in two pilot versions, with previous versions of the game or protocol. 9 participants were excluded due to a breach in protocol or technical error (such as freezing of the tablet due to an imperfect kernel version or children refusing to continue the interaction). Additionally, children with special needs were encouraged to participate but were not included in the analysis ($N=8$). In the end, 25 participants per condition were included ($N=75$; age: $M=9.4$, $SD=0.72$; 37 female). The remaining 13 children who had consent to participate interacted in pairs or alone as we wanted to ensure that all children were able to interact with the robot within the time-frame the school provided and keep a balanced number of participants per condition.

6.4.2 Teacher

The teacher was a psychology PhD student from the University of Plymouth, with limited knowledge of machine learning but with an understanding of human cognition. She had been instructed on how to control the robot using the GUI and the effects of each button. She was aware of the methodology used in the study but not the underlying implementation. She experimented controlling the robot in two interactions: one with the author interacting with the robot and one pilot interaction with a child (not included in the results analysis) to get used to the interface and controlling the robot. After these two interactions, the algorithm was reset and the teacher started to supervise the robot for the supervised condition. No information about the learning algorithm or

the representation of the state and no feedback about the optimal way of interacting or on her policy was provided before or during the study. As such, this study involved, as teacher, a naive user not expert in Machine Learning (ML) and more similar to the general population of expected robot users than an expert in computing.

6.4.3 Conditions

The study evaluated three conditions: the passive condition, supervised condition and autonomous condition. The conditions only impacted the robot's behaviour during the game. In each other part of the study (introduction, tutorial, tests and results), the robot's behaviour was identical between conditions. The study design was between participants, so each participant played four rounds of the game with the same condition (passive, supervised or autonomous).

6.4.3.1 Passive Condition

The *passive* condition served as a control condition. We decided to use a robot even in the control condition to prevent confounds related to the presence of a robot, such as the novelty effect. The goal of the study was not to explore whether a robot can be better than a human or than an interactive touchscreen on its own, but to study whether SPARC can be used by a human to teach the robot a behaviour which has a positive influence on the child during an educational activity. Consequently, in this passive condition, the robot did not provide any advice or feedback to the children during the game, it simply read the instructions and swayed. This condition was used as a baseline for comparison.

6.4.3.2 Supervised Condition

The *supervised* condition was the one in which the robot 'learned', where SPARC was used by a human to teach the robot an efficient policy. As such, it was a dynamic condition where the robot interacted with the child and learned a policy through the human's instructions, while the teacher herself was getting used to controlling the robot. In theory, throughout the different interactions with the children, the behaviour expressed by the robot should be similar and constantly appropriate. In contrast, the teaching interaction between the teacher and the robot should evolve due to the learning component. As the robot learned, its propositions should be more and more appropriate

reducing the requirements on the teacher to select new actions or prevent the robot from executing undesired ones.

6.4.3.3 Autonomous Condition

The *autonomous* condition was used to evaluate whether the learned behaviour was efficient and if the robot could display a useful social policy without being supervised. During this condition, the robot used the policy taught in the supervised condition, but without the teacher in the loop. As such, this condition had to be run after the supervised one, when the teaching was completed.

In the autonomous condition, the robot directly used the suggestions from the algorithm and executed them with a probabilistic delay between 0 and 1.5 seconds based on the teacher's delay in answering the robot's propositions. This delay aimed to give a pace and synchronisation of actions similar to the ones exhibited by the teacher when reacting to the robot's suggestions.

6.4.4 Protocol

At the start of the interaction, a child was first introduced to the robot and told that they would play a game about food chains with the robot. Then, they completed a quick demographic questionnaire and a first pre-test to evaluate their baseline knowledge (cf. Figures 6.7b to 6.7e and Section 6.3.2). After this test, and before starting the teaching game, the child had to complete a tutorial where they were introduced to the mechanics of the game: animals have life and have to eat to survive and the child can move animals to make them interact with other animals or plants and replenish their energy (cf. Figures 6.7f and 6.7g). The teacher sat with the child through these steps to provide clarification if needed. After this short tutorial, the teacher sat away from the child to supervise the robot if required while the child completed two rounds of the game where the robot could provide feedback and advice depending on the condition they were in (cf. Figure 6.7h to 6.7k). Following these initial rounds of the game, the child completed a mid-test before playing another two rounds of the game and completing a last post-test to conclude the study. Figure 6.7 presents examples of screenshot of the Sandtray throughout the interaction.

6.4. METHODOLOGY

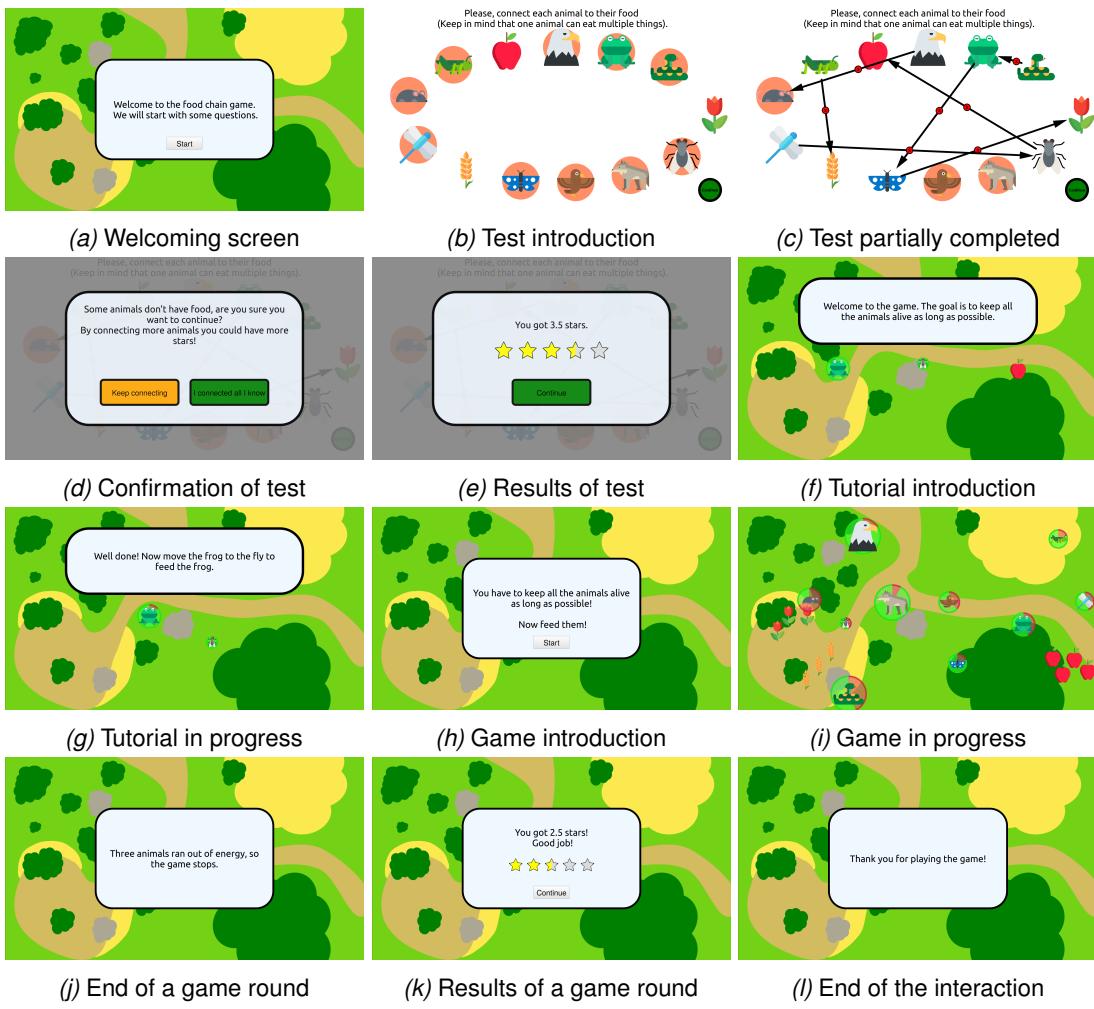


Figure 6.7: Presentation of different steps in the interaction. It should be noted that steps (b) to (e) correspond to one test and (h) to (k) to one round of the game. As such, a full interaction would see a first pretest (steps b to e), followed the tutorial (steps f and g), two rounds of the game (2 repetitions of steps h to k), a second test, two other rounds of the game and a last posttest before ending the interaction with step (l).

6.4.5 Metrics

To address the hypotheses presented in Section 6.2, we collected multiple metrics on both interactions (teacher-robot and robot-child). First, we recorded the actions executed by the robot in the supervised and autonomous conditions to characterise the two policies. Second, we collected two groups of metrics to evaluate the application interaction: the learning metrics (corresponding to the child's performance during the test) and the game metrics (corresponding to the child's behaviour within the game). And finally, in the supervised condition, we recorded the origin of the action executed by the robot (teacher vs algorithm) and the outcome of the proposed actions (executed vs refused).

6.4.5.1 Policy Characterisation

During the game, the robot had access to 655 actions, which can be divided into seven categories: drawing attention, moving close, moving away, moving to, congratulation, encouragement and reminding rules. Due to this high number of actions, the breadth of the state space (210 dimension) and the complex interdependence between actions and states, precisely characterising a whole policy is non-tractable. Consequently, we decided to use the number of actions executed for each category per child to characterise the policy executed by the robot in the active conditions (supervised and autonomous). While not perfectly representing the policy of each condition (e.g. the timing of actions is missing), this metric offers a proxy to compare these policies.

6.4.5.2 Learning Evaluation

As mentioned in Section 6.3.2, the children's knowledge about the food web was evaluated through a graph where children had to connect animals to their food. There were 25 correct connections and 95 incorrect ones. As the child could connect as many arrows as desired, the performance was defined as the number of correct arrows above chance (for the number of connected arrows on the test) divided by the maximum achievable performance. This resulted in a score bounded between -1 and 1. For example, if a child connected 5 good arrows and 3 bad, their performance would be:

$$P = \frac{\#good - (\#good + \#bad) \cdot \frac{totalgood}{total}}{totalgood - totalgood \cdot \frac{totalgood}{total}} = \frac{5 - (5 + 3) \cdot \frac{25}{25+95}}{25 - 25 \cdot \frac{25}{25+95}} = 0.168 \quad (6.1)$$

The three tests (pre, mid and post interaction) resulted in three performance measures. To account for initial differences in knowledge and the progressive difficulty to gain additional knowledge, we computed the learning gain as the difference between the final and initial knowledge divided by the 'progression margin': the difference between the maximum achievable performance and the initial performance. This learning gain indicates how much of the missing knowledge the child managed to gain from the game.

6.4.5.3 Game Metrics

Different metrics were gathered during the rounds of the game to characterise the children's behaviours:

- **Number of different eating interactions:** number of unique eating interactions between two items ([0,25]).
- **Points:** cumulated sum of animals' energy over the game (typical range [550,1100]).
- **Interaction time:** Duration of game rounds, how long a round lasted until three animals ran out of energy (typical range 0.5 to 3 minutes).

As mentioned in the previous section, the children had to explore a food net with 25 good connections and 95 incorrect connections. Due to the imbalance between these numbers, more knowledge is acquired by discovering one of these 25 good connections rather than disproving one of the 95 incorrect ones. As such, we defined our first game metric as the number of different eating interactions children encountered during each game. An eating interaction happens when an animal is moved to its food (or to a predator); and the number of different eating interactions represents how many different unique correct connections the child has been exposed to during the game (multiple eating actions between the same animals would count only once). A game with a high number of different eating represents a game where the child had the opportunity to learn many correct connections between animals. Consequently, by increasing this number, the children would be exposed to more learning items which should help them perform better on the tests. For simplicity, we termed this metric 'exposure to learning items' as it encompass how much knowledge a child has been exposed to in one round of the game. We would expect that an active robot would be able to guide the child towards these correct connections, allowing the child to reach a higher exposure, which would lead to more gain from the game and better overall learning.

On the other hand, both the interaction time and the number of points reached in the game provide information about the children's performance in the task (keeping the animals alive as long as possible) and their engagement. A child disengaged with the game would not play seriously and would reach a lower number of points and a shorter interaction time. We expect that an active robot would encourage and support the child in the learning game and allow them to reach better scores in these engagement and performance metrics.

6.4.5.4 Robot Learning

In the supervised condition, the robot executed actions selected or validated by the teacher, and by using SPARC, the robot could propose actions to the teacher. Faced with a proposition, the teacher had multiple ways to react. They could accept the action (by waiting for it to be executed, pressing ‘Do it’ or selecting the same action manually) or refuse it (by pressing the ‘Cancel’, ‘Skip’ or ‘Remove’ buttons). As such, actions going through this pipeline can be divided in three categories: actions selected by the supervisor, robot’s accepted propositions and robot’s refused propositions. The evolution of these categories represents how much the online component of the learning improved the quality of the robot’s suggestions and reduced the workload on the teacher.

Sometimes the teacher cancelled or skipped actions and then selected them again, we called this effect the ‘reselection’. To obtain the final numbers of accepted and refused actions, we removed these reselections from the refused propositions and we added them to the accepted propositions as it represented cases where the teacher refused an action by accident.

6.5 Results

Similarly to Chapter 5, we analysed the results using Bayesian statistics and the JASP software (JASP Team, 2018). We used a Bayesian mixed ANOVA as an omnibus test to explore the impact of conditions or repetition on the metrics. Additional post-hoc tests used a Bayesian mixed ANOVA comparing the conditions one by one and fixing the prior probability to 0.5 to correct for multiple testing.

6.5.1 Example of a Session

Table 6.4 presents an example of the first minute of a round. Propositions from the robot are in blue, and actions from the teacher in orange. For example, at t=16.9, the teacher accepted the proposition from the robot. Alternatively, in some cases, such as the suggestion at t=20.6, the teacher did not evaluate actions proposed by robot, but just selected another action. In that case, the action proposed is not evaluated and only the selected action is executed and used for learning. In other cases, such as at t=6.6, the algorithm suggested an action, the teacher decided to wait, before selecting it again after a short delay. This is an example of ‘reselection’ as mentioned before. Finally, at t=44.4 seconds, the teacher selected the action to move the mouse closer to the wheat,

Table 6.4: Example of events during the first minute of the first round of the interaction with the 23rd child in the supervised condition. Lines in blue represent propositions from and the robot and orange the reaction from the teacher. ('mvc' is the abbreviation of the move close action)

Time	Event	Time	Event
4.1	childtouch <i>frog</i>	34.4	childtouch <i>wolf</i>
4.3	failinteraction <i>frog wheat-3</i>	34.7	robot proposes remind rules
4.9	animaleats <i>frog fly</i>	35.0	animaleats <i>wolf mouse</i>
5.8	childrelease <i>frog</i>	36.0	teacher selects wait
6.6	robot proposes congrats	36.0	animaleats <i>wolf mouse</i>
7.6	childtouch <i>fly</i>	37.2	childrelease <i>wolf</i>
7.6	teacher selects wait	37.7	childtouch <i>grasshopper</i>
8.0	animaleats <i>fly apple-4</i>	38.3	robot proposes congrats
8.3	childrelease <i>fly</i>	41.8	reset
9.1	teacher selects congrats	42.1	failinteraction <i>grasshopper apple-1</i>
9.1	childtouch <i>frog</i>	42.7	childrelease <i>grasshopper</i>
10.3	childrelease <i>frog</i>	42.7	failinteraction <i>grasshopper apple-1</i>
10.8	childtouch <i>frog</i>	44.4	teacher selects mvc <i>mouse wheat-1</i>
11.2	animaleats <i>frog fly</i>	44.6	robottouch <i>mouse</i>
12.4	failinteraction <i>frog apple-2</i>	44.7	childtouch <i>butterfly</i>
12.5	animaleats <i>frog fly</i>	45.1	failinteraction <i>butterfly wheat-2</i>
13.2	childrelease <i>frog</i>	45.6	childrelease <i>wheat-1</i>
14.2	childtouch <i>fly</i>	45.6	robotrelease <i>mouse</i>
14.5	animaleats <i>fly apple-2</i>	45.7	robottouch <i>mouse</i>
14.6	robot proposes encouragement	48.9	robotrelease <i>mouse</i>
15.0	childrelease <i>fly</i>	49.3	childtouch <i>butterfly</i>
15.4	animaleats <i>fly apple-3</i>	49.3	failinteraction <i>butterfly wheat-1</i>
16.9	teacher selects encouragement	49.6	childrelease <i>butterfly</i>
18.2	childtouch <i>snake</i>	50.0	childtouch <i>mouse</i>
18.4	failinteraction <i>snake wheat-3</i>	50.3	animaleats <i>mouse wheat-1</i>
18.7	animaleats <i>snake bird</i>	51.0	childrelease <i>mouse</i>
19.6	animaleats <i>snake bird</i>	51.1	animaleats <i>mouse wheat-2</i>
20.5	childrelease <i>snake</i>	51.4	robot proposes congrats
20.6	failinteraction <i>snake wheat-4</i>	52.3	teacher selects congrats
20.6	robot proposes congrats	52.9	childtouch <i>snake</i>
20.9	childtouch <i>eagle</i>	52.9	failinteraction <i>snake wheat-3</i>
21.1	animaleats <i>eagle bird</i>	53.2	childrelease <i>snake</i>
22.0	animaleats <i>eagle bird</i>	53.5	childtouch <i>mouse</i>
22.4	childrelease <i>eagle</i>	53.6	animaleats <i>mouse wheat-3</i>
23.3	animaldead <i>bird</i>	54.4	robot proposes congrats
23.4	teacher selects mvc <i>dragonfly fly</i>	54.5	animaleats <i>mouse wheat-4</i>
23.6	robottouch <i>dragonfly</i>	55.0	childrelease <i>mouse</i>
26.9	robotrelease <i>dragonfly</i>	55.6	childtouch <i>dragonfly</i>
27.7	childtouch <i>fly</i>	56.1	teacher selects wait
28.0	childrelease <i>fly</i>	56.8	failinteraction <i>dragonfly apple-1</i>
28.4	childtouch <i>dragonfly</i>	57.3	childrelease <i>dragonfly</i>
28.6	failinteraction <i>dragonfly apple-1</i>	57.5	failinteraction <i>dragonfly apple-1</i>
29.1	childrelease <i>dragonfly</i>	58.6	childtouch <i>grasshopper</i>
29.4	failinteraction <i>dragonfly apple-1</i>	58.6	failinteraction <i>grasshopper apple-1</i>
30.3	childtouch <i>dragonfly</i>	58.8	childrelease undefined
30.3	failinteraction <i>dragonfly apple-1</i>	59.1	childtouch <i>dragonfly</i>
30.7	robot proposes encouragement	59.1	failinteraction <i>dragonfly apple-1</i>
31.0	failinteraction <i>dragonfly apple-1</i>	59.2	failinteraction <i>grasshopper apple-1</i>
31.8	teacher selects wait	59.9	failinteraction <i>dragonfly apple-1</i>
32.5	childrelease <i>dragonfly</i>	60.3	childrelease <i>dragonfly</i>

and after the robot moved the mouse, the child tried other animals and then fed the mouse with the wheat, demonstrating how the actions from the robot could help the children to discover new connections between animals.

6.5.2 Comparison of Policy

Figure 6.8 and Table 6.5 present the number of actions of each type executed by the teacher (in the supervised condition) and by the autonomous robot. Both policies presented similarities: the action ‘Move away’ was almost never used, ‘Move to’ was never used, and the supportive feedback (‘Congratulation’ and ‘Encouragement’) were used more often than ‘Remind rules’ or ‘Drawing attention’. However, some dissimilarities were also present, for instance, the autonomous robot used more encouragements than congratulations while the teacher did the opposite. The autonomous robot also reminded the child of the rules more often and used the ‘Move close’ action less than the supervisor. These differences of actions are probably linked to the type of machine learning used; with instance-based learning, some datapoints will be used in the action selection more often than others, which might explain these biases. But these results show that the robot managed to learn a social and technical policy presenting similarities with the one displayed by the teacher, providing partial support for H2.

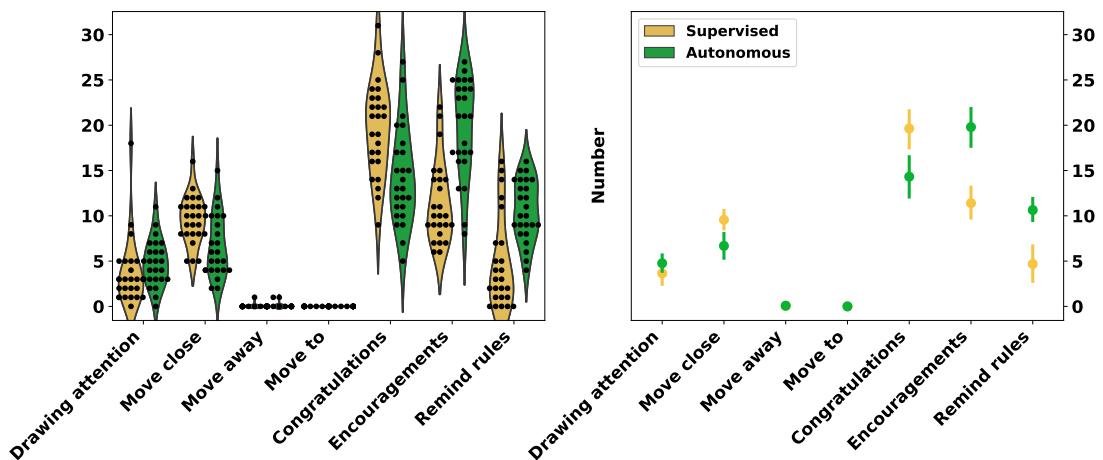


Figure 6.8: Comparison of distribution of actions executed by the robot in the autonomous and supervised conditions. The left graph is a violin plot of the data, while the right presents the means and the 95% Confidence Intervals.

6.5.3 Test Performance

Figure 6.9 and Table 6.6 show the evolution of children’s performance across the three tests. A Bayesian mixed-ANOVA showed that in all conditions, children’s performance increased across the tests ($B = 1.5 \times 10^{12}$). However the impact of the condition on learning

6.5. RESULTS

Table 6.5: Means (SD) of the number of actions executed per child for the active conditions.

	Drawing attention	Move close	Move away	Move to	Congratulations	Encouragements	Remind rules
Supervised	3.6 (3.7)	9.6 (2.6)	0.0 (0.2)	0.0 (0.0)	19.6 (5.1)	11.4 (4.4)	4.7 (5.0)
Autonomous	4.8 (2.6)	6.7 (3.4)	0.1 (0.3)	0.0 (0.0)	14.3 (5.4)	19.8 (5.4)	10.6 (3.3)

was inconclusive with a tendency to show no impact ($B = 0.539$). This indicates that by being involved in the task, every children learned and improved their performances on the test (by gaining on average 13% of the missing knowledge), but the robot behaviour during the game did not have an meaningful impact on the children's learning gain as measured by the test (see Figure 6.10), failing to support H3.

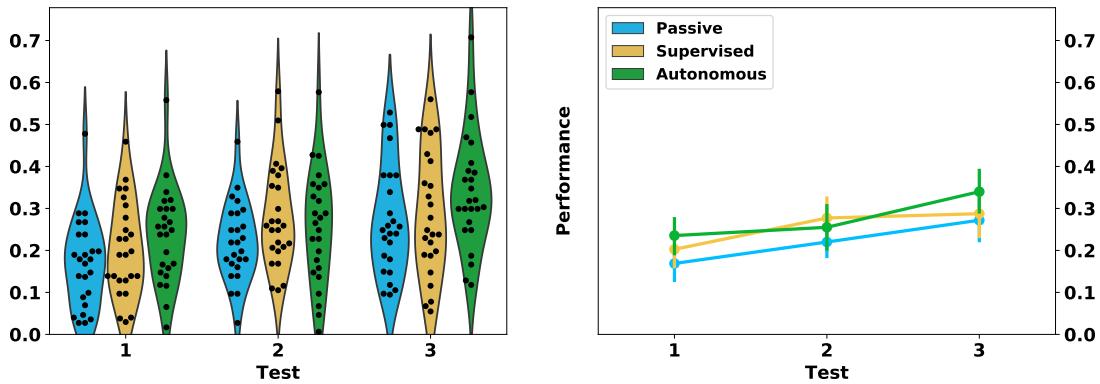


Figure 6.9: Children's performance for the three tests: pretest, midtest and posttest for the three conditions.

Table 6.6: Means (SD) of the children's performance in each test.

	Test 1	Test 2	Test 3
Passive	0.17 (0.11)	0.22 (0.09)	0.27 (0.13)
Supervised	0.2 (0.11)	0.28 (0.12)	0.29 (0.15)
Autonomous	0.24 (0.11)	0.26 (0.13)	0.34 (0.14)

6.5.4 Game Metrics

Different eating behaviours Figure 6.11 and Table 6.7 show the evolution of the number of different eating interactions exhibited by the children across the four game rounds. A Bayesian mixed-ANOVA showed an impact of the condition on the number of different eating interactions produced by the children in the game ($B = 6.1$). Post-hoc tests showed the absence of difference between the supervised and the autonomous conditions ($B = 0.154$), whilst differences were observed between the supervised and

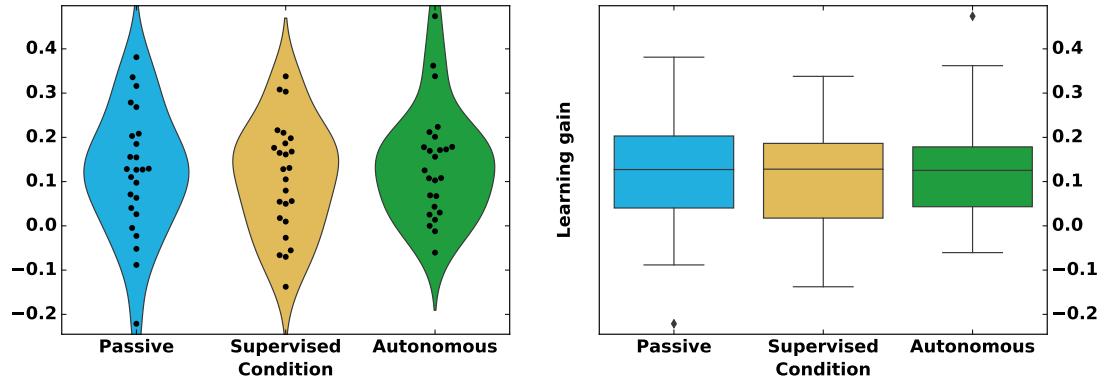


Figure 6.10: Children's normalised learning gain after interacting with the robot for the three conditions.

the passive condition ($B = 512$) and between the autonomous and the passive conditions ($B = 246$). This indicates that, compared to the passive robot, the supervised robot provided additional knowledge to the child during the game, allowing them to create more useful interactions between animals and their food, receiving more information from the game potentially helping them to learn. The autonomous robot managed to recreate this effect without the presence of a human in the action selection loop.

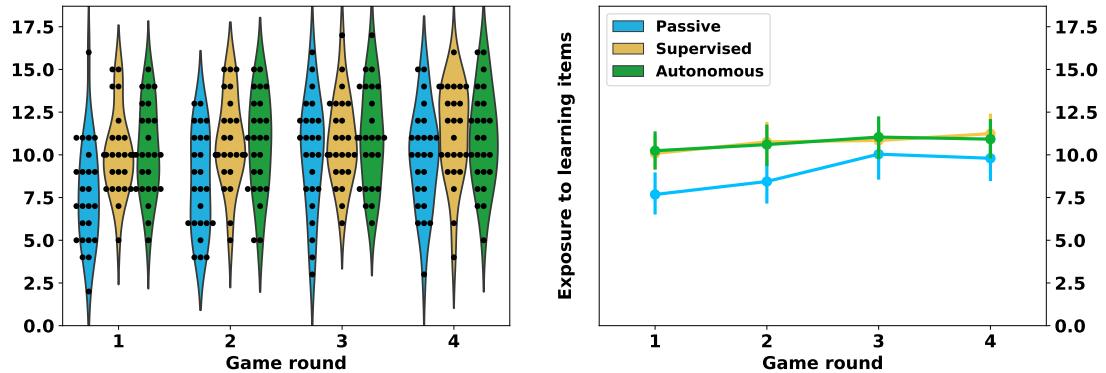


Figure 6.11: Number of different eating interactions produced by the children (corresponding to the exposure to learning items) for the four rounds of the game for the three conditions.

Table 6.7: Means (SD) of the number of different eating interactions produced by the children in each round of the game.

	Round 1	Round 2	Round 3	Round 4
Passive	7.7 (3.0)	8.4 (2.9)	10.0 (3.3)	9.8 (3.0)
Supervised	10.1 (2.5)	10.8 (2.6)	10.8 (2.5)	11.2 (2.8)
Autonomous	10.2 (2.7)	10.6 (2.9)	11.0 (2.9)	10.9 (2.9)

Points Figure 6.12 and Table 6.8 show the evolution of the number of points achieved by the children across the four game rounds. A Bayesian mixed-ANOVA showed an impact of the condition on the number of points achieved by the children in the

6.5. RESULTS

game ($B = 10.0$). Post-hoc tests showed a strong difference between the passive and the supervised conditions ($B = 5.1 \times 10^4$) and differences between the supervised and the autonomous conditions ($B = 5.2$) and the autonomous and the passive condition ($B = 5.9$). This indicates that when the robot was supervised, it allowed children to achieve more points than a passive robot. A similar effect was observed when the robot was autonomous, however the autonomous robot was less efficient than the supervised robot in helping the children to achieve a high score in the game.

The repetitions (i.e. the number of rounds complete before) tended to have an impact on the children's score in the game ($B = 2.4$) with differences in the post-hoc test only between the rounds 1 and 2 ($B = 3.2$) and the rounds 1 and 4 ($B = 5.0$). This demonstrates a practice effect where children achieved a better score in the last round than in the first.

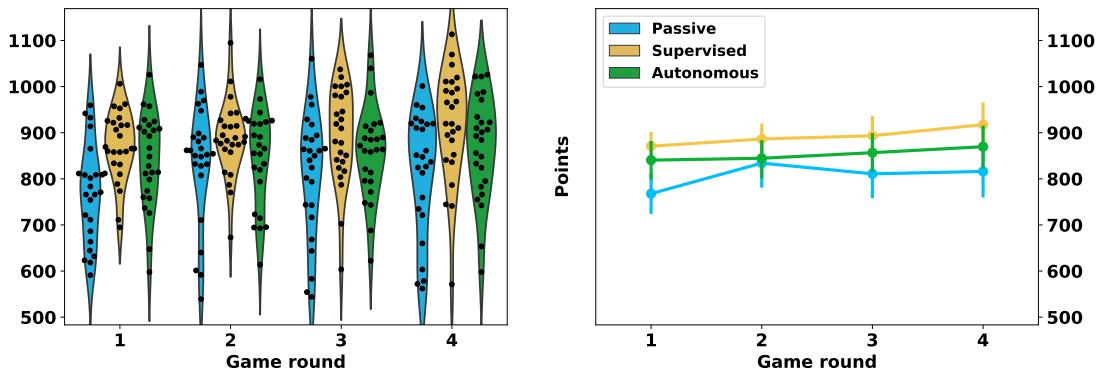


Figure 6.12: Points achieved by the children in each round of the game for the three conditions.

Table 6.8: Means (SD) of the number of points achieved by the children in each round of the game.

	Round 1	Round 2	Round 3	Round 4
Passive	767.8 (105.3)	834.1 (127.0)	810.8 (133.1)	816.0 (132.6)
Supervised	870.8 (75.3)	886.3 (81.6)	893.4 (104.7)	917.4 (120.0)
Autonomous	840.6 (101.3)	844.6 (103.3)	856.7 (99.7)	869.6 (111.9)

Time Figure 6.13 and Table 6.9 show the evolution of interaction time across the four game rounds. A Bayesian mixed-ANOVA showed inconclusive results on the impact of the condition on the interaction time in the game ($B = 1.05$). Post-hoc tests showed no difference between the supervised and the autonomous conditions ($B = 0.287$), however, differences were observed between the supervised and the passive condition ($B = 118$) and a trend towards a difference between the autonomous and the passive conditions ($B = 2.9$). This indicates that children were better at the game in the supervised condition

whereby animals were alive longer than in the passive condition. The autonomous robot learned and applied a policy tending to replicate this effect and without exhibiting differences with the supervised one.

However, the analysis showed no effect of the repetitions on the interaction time ($B = 0.023$). The children did not manage to keep the animals alive longer with more practice at the game. One of the reasons was a partial ceiling effect at 2.25 minutes (see the dashed red line on Figure 6.13). When not fed, animals would run out of energy at 2.25 minutes, so if children did not manage to feed most of the animals at least once before that time, the game would stop.

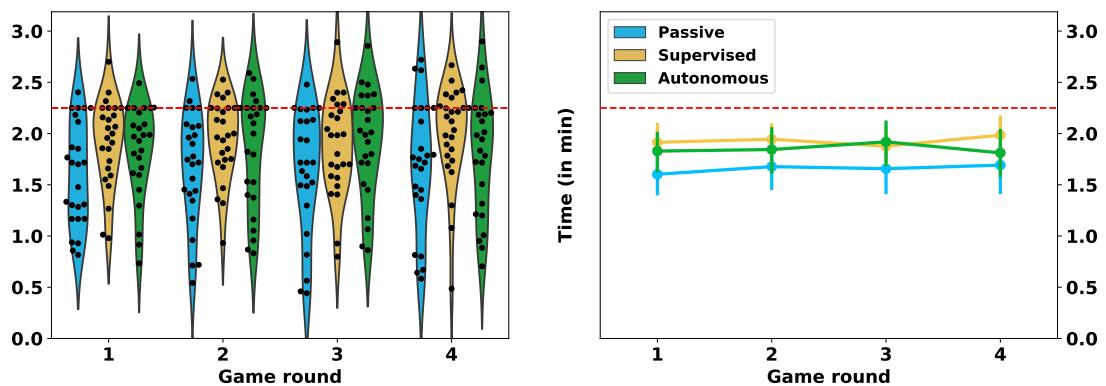


Figure 6.13: Interaction time for the four rounds of the game for the three conditions. The dashed red line represents 2.25 minutes, the time at which unfed animals died without intervention, leading to an end of the game if the child did not feed animals enough.

Table 6.9: Means (SD) of the duration (in minutes) of each round of the game.

	Round 1	Round 2	Round 3	Round 4
Passive	1.6 (0.5)	1.68 (0.54)	1.66 (0.6)	1.69 (0.63)
Supervised	1.91 (0.42)	1.94 (0.39)	1.88 (0.48)	1.98 (0.49)
Autonomous	1.83 (0.46)	1.84 (0.55)	1.92 (0.52)	1.81 (0.58)

Summary These game metrics suggest that the policy executed by the autonomous robot allowed children to achieve similar results in the game to the supervised robot, and better results than when interacting with a passive robot. This provides support for H2 ('The autonomous robot will be able to interact socially and efficiently during the game and maintain the child's engagement during the learning task.').

6.5.5 Teaching the Robot

Figure 6.14 presents the reaction of the teacher to the robot's suggestions across all the supervised interactions. Contrary to our expectations, the number of accepted and

refused suggestions as well as teacher selections stayed roughly constant throughout the interactions with the children (as the teacher aspect was a case study with a low number of datapoints and high variation between children, inferential statistical analysis, such as regression, would not be appropriate). On average, among the 4 rounds of an interaction, the teacher accepted 17.2 ($SD=4.0$) actions proposed by the robot (including selections of proposed actions and reselections - cf. Section 6.4.5) and 41.7 ($SD=11.1$) were refused by the teacher per interaction. The teacher manually selected 25.8 ($SD=5.8$) actions per interaction. On average, 29.2% of the robot suggestions were accepted, while 70.8% were rejected. We would have expected these results to be different: with the learning, the number of accepted propositions should have increased and both the number of refused propositions and teacher selections should have decreased. These unexpected results are discussed in details in Section 6.6.2.2. It should also be noted that, due to the case study effect, these results might not be replicated with another teacher.

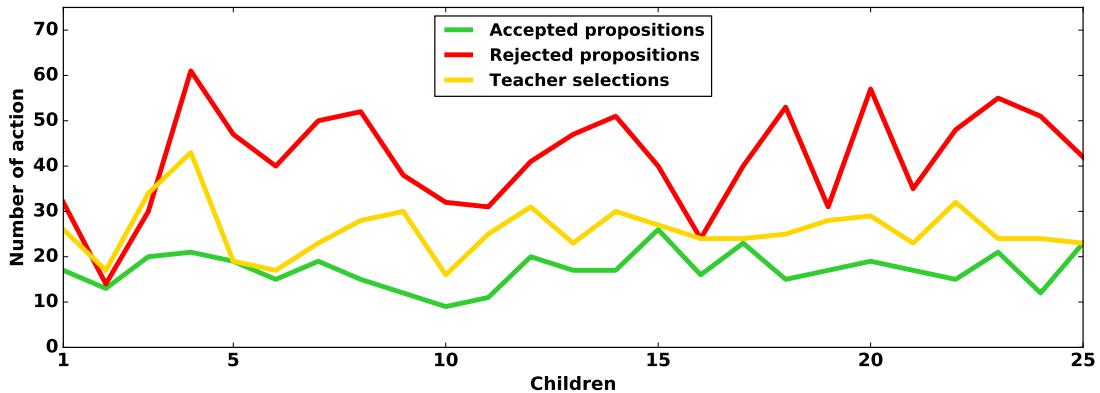


Figure 6.14: Summary of the action selection process in the supervised condition: the ‘teacher selection’ label represents each time the teacher manually selected an action not proposed by the robot.

Figure 6.15 presents the accumulated number of different actions the teacher used. This number represents the complexity of the policy used by the teacher and shows that across all of the interaction, the teacher tended to demonstrate all the useful actions, reaching a final plateau towards the end of the last interaction. We can observe a sharp increase in the first 5 interactions, when the teacher used the main actions for the first time. Then there is a mixture of small plateaus and small increases, indicating that the teacher alternated phases where she enriched her policy and phases where she maintained her policy. And finally, the number of different actions in the policy seems to converge around 56 towards the last interactions. This indicate that as intended,

the action space is general: only a subset of it (56 actions out 655) is relevant to this task, for another task or another desired robot behaviour, a different subset of the action space could have been useful. By using human demonstrations, the robot can learn which actions are important for the current task without having to explore the whole action space. This non reliance on exploration allows the robot to learn a policy without being impacted by the dimension of the general action space. We precise that this figure represents only the number of different actions used by the teacher, see Figure 6.8 to have information about the number of actions used for each type.

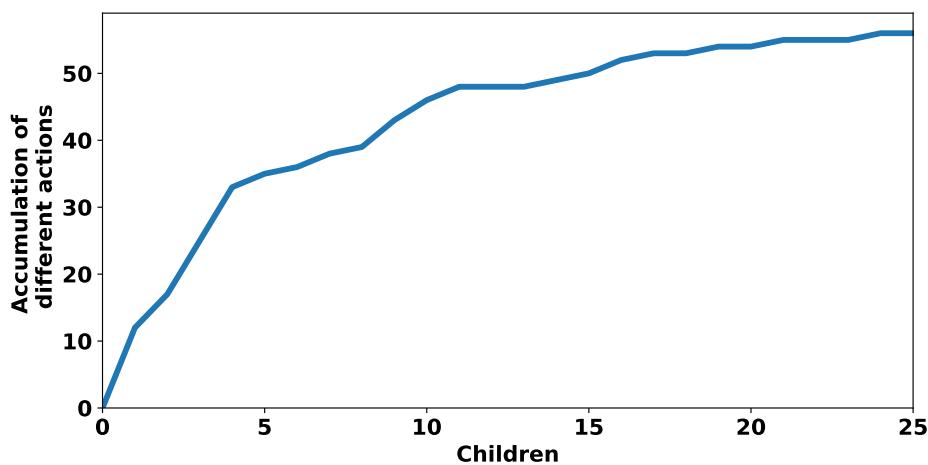


Figure 6.15: Number of different actions used by the teacher throughout the interactions with the children.

In post-hoc discussion, the teacher reported three phases in her teaching:

- First sessions: she was not paying much attention to the suggestions, mostly focusing on having the robot executing a correct policy.
- Sessions 6 to around 16: she was paying more attention to the suggestions without giving them much credit.
- Last sessions: she started to trust the robot more but without ever trusting it totally.

Appendix F presents a more detailed diary of the teacher throughout her supervision. Additionally, while the buttons cancel and skip had different impact on the learning, and were designed to be used in different cases, they had the same visible impact on the robot behaviour: not executing the action. Consequently, the teacher reported that she used them interchangeably, which highlights that despite our efforts, there were some misunderstandings during the description of the interface.

The teacher did report a decrease of workload as she progressed in the sessions number. This was supported by behaviours such as typing her observations on a laptop, while gazing at the interface in multiple interactions (especially at the start of a round). However, as shown by the evolution of curves in Figure 6.15, this decrease of workload seemed to be due mostly to the teacher getting used to the interaction, and not to the online learning and the improvement of the suggested propositions, invalidating H4.

6.6 Discussion

This study explored whether a human could teach a robot a policy to support a child in an educational game. Two statements: “by receiving supervision from a human teacher, a robot can learn while only displaying an appropriate behaviour” and “after learning, such a robot would be able to interact autonomously and efficiently with humans”, were evaluated by comparing three conditions in a study involving an educational activity with 75 children. The passive condition corresponded to a robot not providing any feedback during the game and was used as the baseline. In the supervised condition, a human taught the robot to interact with children using SPARC; and in the autonomous condition, the robot applied the policy learned in the supervised condition to interact with children without supervision.

The main finding of this study is that SPARC allowed a human to successfully teach a robot to interact with a child *in situ*, in a complex and real-world environment. This is to compare with methods such as RL which rely on random exploration and consequently fail to produce an appropriate behaviour during the learning phase, thus preventing their application to HRI. Furthermore, the autonomous robot demonstrated a policy presenting similarities with the one used by the teacher in the supervised condition, and both these policies had positive effects on the child during the game compared to the passive robot (increase of performance: points and time, and exposure to learning items: number of different eating behaviours). However, this increase in positive behaviours during the games did not transform into additional learning gains in the tests. Additionally, when the robot was supervised, the number of actions manually selected by the teacher and the number of proposed actions the teacher accepted and refused remained stable through the different interactions.

This section will first discuss the impacts of this work, the advantages of SPARC when teaching robots a policy while interacting in a sensitive environment. Then, we will

discuss the limitations of the study: the unexpected results and the methodological constraints. Finally, we will present relevant considerations for applying SPARC to HRI.

6.6.1 Impacts

This study demonstrated the first application of SPARC to real-world HRI. Despite the complexity of real-world interaction (multimodality, sociality, high stakes and unpredictability), a robot was successfully taught to interact efficiently with humans in a complex and rich social environment from *in situ* supervision. During her supervision, the teacher applied a social policy taking into account a large amount cues partially available to the robot and social norms, such as turn-taking, when selecting actions. However, due to the complexity to measure such effects, no analysis of the sociability of the autonomous robot has been done. But the autonomous robot's policy and the one demonstrated by the teacher during the supervision presented similarities. Both policies generated comparable distributions of actions and produced similar impacts on the children (better performance in the game and higher exposure to learning items). This supports H2 ('The autonomous robot will be able to interact socially and efficiently during the game and maintain the child's engagement during the learning task.'). This is an important contribution as the robot learned to interact efficiently in a complex, multimodal and social environment in the real world, including large state and action spaces and where errors could have an important cost to the interaction (for instance having a child refusing to continue the interaction). As discussed in the next sections, such a capability would allow robots to be deployed more easily, in more environments and with policies tailored to each situation.

6.6.1.1 Human Control

By analysing the exact actions executed by the robot in both the supervised and the autonomous conditions, we observed that the robot only executed actions relevant to the current game. For example, the robot only moved animals close together if they were related (only moving an animal to appropriate food items and not towards the other items). Similarly, as demonstrated by Figure 6.8, the actions 'move away' and 'move to' were rarely or never used, possibly because they could be confusing for the child. Figure 6.14 also indicates that the teacher did use the 'Skip' and 'Cancel' buttons to prevent undesired actions being executed by the robot. Consequently, H1 ('In the supervised condition, the teacher will be able to ensure an appropriate robot behaviour

whilst teaching.') is supported. By having the robot submitting actions to its teacher before executing them, SPARC allows teachers to ensure that the robot's behaviour is appropriate even during the learning phase, thus enabling robots to learn *in situ* in complex and sensitive environments.

Furthermore, in her diary (cf. Appendix F), the teacher identified herself as the agent making the decisions concerning the robot's actions (and not as someone having to deal with the consequences of an uncontrollable agent's decisions) and used verbs related to control to express her behaviour: "I used", "I allowed", "[I] let the robot carry out", "I gave", "[I] chose" or "[I] added". She further mention that "It's now fairly easy to control the robot" or "Controlling the robot is really easy now, although I still tend not to let it carry out its suggested actions even when they are valid." Not only is H1 supported, but the teacher diary also shows that the teacher felt in control and that control got easier with experience. Whilst we cannot conclude about user experience, this does at least show promise for future use and research.

6.6.1.2 Learning *in situ*

One of the main features of SPARC is that it can be used for learning *in situ*. This learning in the real world has multiple advantages compared to learning in simulation or simply manually designing controllers before deployment. First, the controller is fed real data from the interaction and can specialise its policy to the current task and partners specificities. This is especially notable as human behaviours can be unpredictable, and as such, interaction in the real world might differ highly from expectations. Secondly, it means that the same robot with the same learning algorithm and representation of the world can learn different policies depending on the variations in the environment and the teachers' goals and desires. Therefore, robot's behaviour might be more precisely tailored to each application and its requirements. Lastly, even when learning, the robot displays a useful behaviour, thus the time required for the robot to be taught a policy is directly put to use in the desired environment. For example in the context of education, the robot can already tutor children while it is learning how to interact. As demonstrated by children's behaviours in the supervised condition, the robot learning does not perturb the children, in contrast, even during the robot's learning phase, the children benefited from the robot's presence.

This positive impact of the robot behaviour even in the teaching phase is a central characteristic of SPARC and is only achievable with the presence and control of a human supervisor. Other methods that learn *in situ*, such as RL (Sutton & Barto, 1998), present the same two first advantages (learning from real data and adaptivity), but due to the reliance on exploration, the robot's behaviour during the learning phase is not always appropriate, limiting the application of such methods to HRI or other sensitive environments.

6.6.1.3 Teaching a Robot to Interact

Even though the online learning did not reduce the teacher's workload much during the interactions, SPARC still possesses two advantages compared to offline learning from WoZ (Sequeira et al., 2016) or from pure human demonstrations (Liu et al., 2014). Firstly, the learning algorithm has access to more datapoints: the teacher's selections and their reactions to the algorithm's propositions. Classical Learning from Demonstration (LfD) only has access to demonstrations, thus SPARC provides more datapoints to the learning and enables progressive refinement if the robot's policy. Secondly and more importantly, the learning process is more transparent. By receiving and evaluating the robot's propositions, the teacher can estimate the robot's knowledge and create a mental model of the robot's policy. This continuous communication between the teacher and the robot increases the transparency of the learning process, and might allow the teachers to build trust grounded by their experience with the robot, knowing its strengths and weaknesses. This transparency can ease the decision of deploying the robot to interact autonomously as the teacher has experienced the policy and knows what to expect from the robot.

6.6.1.4 Ease of Access

This study also used a teacher with limited knowledge in robotics and ML and unaware of the state representation or the algorithm involved in the robot learning. Despite this limited technical knowledge, by following simple guidelines this teacher succeeded in controlling the robot and providing it with an efficient policy. This supports the argument that SPARC is more accessible to the general population and by extension might allow more people to teach robots complex behaviours in the real world. This has major benefits in that if people do not need to know robotics or how to code in order to teach a robot a social behaviour, methods such as SPARC may help to democratise the use of

robots. Robot designers would only have to prepare teachable robots and the users would be able to craft a policy fulfilling their specific needs, thus both reducing the workload on the engineers designing robots and improving robots' usefulness for the general public.

6.6.1.5 Generalisation

This study demonstrated that SPARC can be used to teach a robot a precise and rich policy in a complex environment. The state spaces contained 210 continuous dimensions and the action space 655 discrete actions and both of them were multimodal. But despite the complexity of this world representation, the algorithm reached a policy which had similar effects on children as a human tele-operating the robot.

This capacity to learn a precise policy from a large representation of the world is key in HRI. As mentioned in Section 6.3.6, to be able to demonstrate rich behaviours, social robots need to have representations of the world containing many dimensions, covering different situations. However, the increase in dimensionality of the state and action space might lead to an exponential increase in the number of datapoints required to learn an efficient policy. This effect is called the curse of dimensionality (Bellman, 1957) and makes it more difficult to learn in such large spaces, especially when datapoints have to come from the real world (gathering datapoints by interacting with humans can be time consuming, expensive and potentially risky for the partners involved in the interaction).

By using human teachers to guide the robot through this complex space, SPARC enables robots to learn rich policies in complex environments and from a limited number of datapoints. In this study, by slicing the state space and using demonstrations, the initial dimensions of the state and action spaces are made irrelevant; adding dimensions to the space will not impact the learning if the teacher does not select them. The state could be as large as desired, yet as long as the teacher has a way to inform the algorithm of the relevant features, the robot can learn an policy quickly.

Being able to learn in generic spaces has multiple advantages. First it means that from a single state representation many different behaviours can be taught. For example, instead of creating a supportive robot, the teacher should be able to create an adversarial or a playful robot behaviour. Additionally, as argued in Section 6.3.5.3, it allows for the world representation to be repurposed more easily than static, simple representations

tailored to a specific task and goal. For instance, we would expect that only a limited amount of work would be required to change the setup for another task involving moving images (such as maths or language in the context of education), and the robot could be taught a new policy adapted to this new environment with the same algorithm and a similar state and action space.

6.6.2 Limitations of the Study

Despite showing positive results, this study also suffered from some limitations. The next two sections describe potential reasons for the unexpected results: the absence of differences between conditions in learning gain and the lack of increase in the number of actions proposed by the robot accepted by the teacher. The last part highlight the limitations of the protocol: the impacts of the teacher and the stopping criteria on the results observed in this study.

6.6.2.1 Children's Behaviours and Learning Gain

The active robots' behaviour encouraged children to produce more 'useful' behaviours during the interaction. When interacting with the supervised or autonomous robot, children were engaged more actively in the learning activity (as demonstrated by better performance in the game and higher numbers of different eating interactions exhibited by children in these conditions compared to the passive one). However, this higher engagement in the activity and exposure to learning items did not translate into an increase in learning gain in the test. In the three conditions, the children had similar test scores; thereby not supporting H3 ('An active robot (supervised or autonomous) supports child learning: the learning gain in the passive condition will be inferior to the learning gain in the autonomous condition, which will be inferior to the learning gain in the supervised condition'). We identified two possible causes for this absence of transfer between game behaviours and test performance.

Limitations of the game. One potential explanation is that the game by itself, without the robot, encouraged the children to explore, and interacting independently with it was sufficient to promote learning. As such, in the active conditions, the robot's behaviour might have distracted children from their exploration or encouraged them to rely on or to attend to the robot rather than their own exploration. In that case two effects might have cancelled each other out: on one hand the behaviour of an active robot provided

additional knowledge to the children, but on the other hand the same behaviour might have perturbed children's independent exploration of the game, potentially reducing their learning. This might explain the absence of any effect of the robot's behaviour on the children's learning gain as measured by the test. Additionally, as demonstrated by the low improvement in children's performance in the game, the game might have been too difficult for this age range (in pilot studies, adults focused on the game easily reached high scores both in the games and the tests, but even some adults have been seen to achieve limited performance). This may be explained by the fact that the game was not validated to show it allows for substantial learning. Consequently, a simpler game but with a longer and shallower learning curve would probably have led to clearer results and potentially greater differences between the conditions.

Limitations of the test. The second possible explanation is that the test itself might not have been able to capture the children's knowledge accurately. The test only asked children to connect as many animals to their food as possible. However it might have been too open-ended, and might not have encouraged children to make all the connections they knew to be correct, but simply to make an arbitrary number of connections. Having forced-choice questions, for instance randomly selecting 20 connections and asking children if they are correct, might have provided a better evaluation of the children's knowledge. It should also be noted that as the test aimed at estimating the children's knowledge and not improve it. The test did not include feedback on the correctness of the children's answer, thereby the children were not given immediate reinforcement for correct connections. In applications where the knowledge gain is more important than its evaluation, the test could also provide additional information to the children, not only evaluating their knowledge, but also supporting their learning. Similarly to the game, the test was not validated to show it allows for substantial capture of the children's knowledge.

Finally, both for the games and the tests, a large disparity was observed within conditions. As the effects of the condition might not be large, it would be important to improve them and replicate the study with other children or with more participants.

6.6.2.2 Robot Learning

As presented in Chapter 3, one of the motivations for SPARC is to provide a way to smoothly move away from WoZ to Supervised Autonomy, potentially leading to pure

autonomy. By learning the policy online, the number of actions selected or corrected by the teacher should decrease and the number of accepted suggestions should increase, hence reducing the teacher's workload. However, this expectation (and by extension H4 - 'Using SPARC, the supervisor's workload decreases over time: the number of corrected actions and the number of actions selected by the teacher decrease with practice, while the number of accepted proposed actions increases') was not validated by this study's observations. The number (and ratio) of actions accepted, refused and selected by the teacher remained stable throughout the interactions. We identified five potential explanations for this effect.

Suggestion rate too high. The first possible explanation is that the robot proposed actions too often (58% more than the total number of executed actions). This indicates that the adaptive threshold restricting actions from being proposed was consistently too low. This high number of proposed actions partially explains the high number of actions refused by the teacher. Unstructured interviews with the teacher after the study revealed that as the robot tended to propose actions at a high frequency, the teacher reached a point in her supervision where she often preferred refusing the robot's propositions even in cases where they were correct rather than taking the time to evaluate them and risking having undesired actions executed. This is also supported by notes in the teacher's diary in Appendix F:

- “I skipped/cancelled a lot of the robot’s suggestions, including ones which I wanted to approve”.
- “I find I’m dismissing robot suggestions more than I actually want to - some are valid but I am ‘playing safe’ by skipping/cancelling all in order to avoid inappropriate suggestions”.
- “robot was often suggest[ing] good things but I was auto-skipping them”.

This issue could be tackled by fine-tuning the algorithm, for example by adapting the factors driving the increase or decrease of the threshold selecting actions to propose. Additionally, the rule updating the threshold might not have been optimal: the threshold might have been decreased in cases where it should not have been and vice versa. Consequently, the conditions to increase or decrease the threshold could be modified by taking into account which action was the closest to the current state not only the

distance with the current one (cf. Section 6.3.6). Alternatively, the filter transferring actions to the teacher could also be adapted to force a minimum time between two suggestions to reduce the load on the teacher. But, as this might limit the range of policies available to the learning algorithm, this would need to be seriously considered before being implemented.

Human adaptation. A second effect which may have limited the correctness of the propositions is the evolution of the teacher's policy. As the teacher progressed in the interactions, she used a wider range of actions (as shown by Figure 6.15). As the algorithm only proposed actions already used at least once, this led to a requirement for the teacher to first demonstrate each action before having the algorithm propose them. This limited the visibility of the learning before the policy was demonstrated in sufficient detail. However, the convergence of the number of actions used by the teacher towards the final interactions indicates that the algorithm might have been better at predicting the teacher's actions if the teaching phase had been extended.

Continuous time. This study also stood out from classic evaluation of ML because instead of existing only in a discrete time (as generally used for an RL agent), here the robot interacted in real-time. In classical Markov Decision Process (MDP) frameworks, an action has to be selected at each step, actions last one step, and optimal strategies might exist. However, when interacting in the real world, actions take many steps and in most time steps no action should be selected. Additionally, due to the continuous aspect of time, actions are not uniquely valid at a specific step, but around that step. This implies that to reduce the teacher's number of selected actions, the algorithm does not have to select the same action as the teacher at each step, but needs to anticipate the teacher's actions, so that the teacher does not select them first. For example, if the algorithm selects a correct action one step after the teacher's selection, this action would simply be filtered out by the action analyser and as such would not be considered as a good proposition. This might contribute to the limited visibility of the results (absence of increase in number of propositions accepted by the teacher), while the actual policy of the autonomous robot was similar to the supervised one and both had matching positive impacts on the children.

Algorithm. Another element potentially explaining the low approval rate of the online learning is related to the algorithm itself. In its simplest form (and as used in this

study), Nearest Neighbours considers only one neighbour, making it highly sensitive to outliers. Consequently, some instances in memory could be used too often, leading to an imbalance of policy compared to the demonstrated one. For example, as shown by Figure 6.8, the action ‘Remind Rules’ is used more often by the autonomous robot than the teacher. This could be due to a few instances of this action in locations of the space visited more often than other actions. Additionally, the structure of the space, and especially the partial continuity of some states (such as time since events) might increase this effect of an instance’s location, resulting in some of them being visited more often than others. One way to tackle this issue could be to use k Nearest Neighbours instead, this might lead to a more robust learning algorithm matching the teacher’s policy more closely.

Difference of state representation between human and robot. As mentioned in Section 2.2.4, Knox et al. (2014) and Sequeira et al. (2016) stress the fact that a human teacher (or demonstrator) should have access to the same features as the algorithm to increase transferability. However, even when this recommendation is met, the teacher can create temporal structures which may not be available to the algorithm. This would lead to the presence of *hidden states* the teacher uses for their selections that the robot does not have access to.

In this study, for methodological reasons, we did not remove the teacher from the room when the child interacted with the robot. Consequently, this allowed the teacher to have access to features of the interaction absent from the state representation used by the algorithm. For example, the teacher reported the following statements in her diary:

- “I try to use participants’ performance on pre and mid tests to inform my teaching”
(This data was not available to the algorithm).
- “Participant 18 - talks about what they’re doing so [it’s] easy to offer direction.
Says things like ‘I don’t know what xx eats’ ” (The audio was not processed by the system).
- “Seemed quite scared when the robot moved so kept demonstrations to a minimum.” (Emotions were not analysed by the system).

These differences between the state of the world used by the teacher and the one available to the algorithm could partially explain some of the limits of the learning. A

way to tackle this issue could be to have a larger state, including more dimensions and relying on the teacher to indicate more precisely which dimension they used.

Summary While the autonomous behaviour was different from the supervised behaviour, both policies still presented many similarities in the distribution of actions executed by the robot and the children’s reactions to the robot’s behaviours. Additionally, the challenge of learning a suitable policy for the robot should be highlighted: the state space was large (210 dimensions), continuous and the action space contained a large number of possibilities. The interaction was also social and multimodal, happened in the real world and involved real humans with a wide range of behaviours, initial knowledge, preferences and abilities. Finally, the algorithm was not provided with any ‘innate’ knowledge, all the dimensions of the state or the actions were treated the same way, the semantics of the interaction were inferred from the demonstrations and feedback. With more training data and an improved algorithm and interface, the robot learning could have been more efficient and might have led to a decrease in the teacher’s workload and an increased similarity between the teacher’s policy and that of the autonomous robot.

6.6.2.3 Protocol Limitations

Teacher. SPARC includes two distinct but simultaneous human-robot interactions. Evaluating such interconnected interactions is a complex task as each human’s behaviours impacts the other’s. To explore in a repeatable and comparable manner one of the interactions, the other one needs to be as consistent as possible. However, humans are not consistent and, consequently, evaluating these two interactions simultaneously is a challenge. By deciding to keep the same teacher for all the interactions, we only have a sample of one participant as a robot teacher. As a result, this study is in essence a case study of one participant teaching a robot to interact with children; and this could have created some biases in the supervision. As seen in previous chapters, different humans would teach the robot differently. It would have been interesting to explore this axis, by introducing distinct teachers and observing how their different behaviours impacted the learning process and the final policy. We would expect that the resulting autonomous behaviours should match each teacher’s policy. However, due to the variability of children, a large number of participants would be required to evaluate a robot

learning from several teachers. As such, we did not evaluate more than one teacher in this study.

Stopping Criteria. During this study, each condition consisted of 25 children. This number was selected in order to have a balanced number of participants in each condition considering the number of children available in the classes visited. This means that the robot learning was terminated at some point without possibility to refine its policy further, and this impacted all the results observed in the autonomous condition. By involving more children in the supervised condition, the teacher's policy might have converged more, the learning algorithm might have increased its performance by suggesting more appropriate actions to the teacher and potentially reducing her workload. This relatively arbitrary cut-off of the learning phase had important effects on both the supervised and autonomous conditions, and continuing with more children or stopping earlier would have probably led to different results. For example, additional interaction with children might have modified the value for the threshold changing the number of proposed actions, potentially leading to the autonomous robot executing more or less actions, which would probably also have impacted the children reactions in the autonomous condition.

6.6.3 Considerations When Applying SPARC

6.6.3.1 Human in the Loop

With SPARC, the human teacher is fundamental, they are the key to both providing the safety and allowing the robot to learn quickly. By having control over the robot, they can ensure that the robot only executes desired actions in the interaction. This has two positive effects on the learning process. First, it provides correct labels for states, thus reducing the need for exploration. Second, by guaranteeing the interaction to follow a normal and efficient flow, it ensures that the robot only interacts in relevant parts of the state. If the robot was learning on its own, it might spend time and effort exploring undesired parts of the environment, which would be detrimental both to the learning and to the interaction partners.

However, by being the cornerstone of SPARC, the human teacher also puts constraint on the interaction. To be able to maximise their efficiency and impact on the learning and the interaction, humans have to stay constantly in control. As introduced in Section 3.5.1 they need to be provided with time to correct all the actions intended by the robot before

their automatic execution. Additionally, they need to be able to constantly select a correct action for the robot to execute. To be a successful teacher, the human needs both to have access to the tools to control the robot, and to use them efficiently.

In this study, while the interface technically allowed the teacher to control precisely the robot's behaviour, this human control was not perfect throughout this study. The teacher reported once that “[she] Ended up accidentally making a mistake in my demonstration” and “[she] kept missing the skip button so a few erroneous behaviours slipped through”. This demonstrates that humans are not perfect, thereby one of the assumptions we made when proposing SPARC (a human teacher in control of the robot behaviour can prevent it to do any undesired actions) was violated. Even when being focused on the task, if the interaction lasts for a sufficient amount of time, no human will be able to behave perfectly, which will lead to errors in supervision. These errors being unavoidable, as argued by Rasmussen & Vicente (1989), interactive systems should be transparent for their users and allow error recovery. The next section will describe more in details how error recovery was provided in the study and will also offer other alternatives.

To be successful when using SPARC, the teacher needs to be provided with an environment where they can reliably exert control over the robot. If this teaching environment is not carefully controlled (e.g. by having a robot suggesting too many actions or by having a correction window too short), the teacher's behaviour will be suboptimal. As such when creating an application for SPARC, designers need to seriously consider the teaching environment provided to the teacher and anticipate potential limitations of the system being evaluated. If the system is correctly designed or at least these limitations are kept to a minimum, the robot learning can be improved, leading to robot behaviour which might have taken a prohibitive amount of time to hand code or be learned autonomously. This highlights the importance of the interface between the teacher and the robot and ways to recover gracefully from errors and ensure that the teacher can teach efficiently.

6.6.3.2 Interface

As pointed out in the section above, when applying SPARC to a situation, the interface used by the teacher to communicate with the robot is key. This interface defines how the teacher can interact with the robot, how much control the teacher possesses, how much

information the teacher has about the learning progress, how much workload will be required to supervise and teach the robot and how the teacher can recover from errors.

With SPARC, the interface needs to allow the teacher to:

- Select any action available to the robot.
- Specify the relevant features to an action to speed up the learning.
- React to and evaluate each proposition of the robot before a potential auto-execution.
- Monitor the learner.
- Recover from errors.

It should be noted that such an interface might require training for the teacher. For example, in this study the interface was explained to the teacher and she trained on some interactions not included in the results. While training the teacher might be required, interfaces used for SPARC should be designed to be as simple as possible.

Select actions. The teacher needs to have access to every required action and be able to select them and have the robot execute them in a timely fashion. However, as seen in this study, the action space can be large (here, around 56 different actions were used and more than 600 were available), consequently, each action might not be representable with an individual button. As such, intuitive ways need to be found to both make actions available to the teacher but also make the interface easy to use (requiring limited training time beforehand and keeping the workload low during the supervision).

In this study, instead of directly having the teacher select one of the 655 actions, the action analyser inferred the desired action by analysing the type of action selected, the relevant animals provided and how the action would impact the distance between animals. While giving access to all the possible actions without requiring a button for each, indirect action selection through inference might lead to interpretations errors, or add special requirements on the way the teacher selects actions to ensure unambiguous interpretation.

Additionally, as some actions would have to be repeated multiple times in the same interaction, to reduce the risk of boredom each action had multiple utterances. While

having been selected to be similar, these different utterances (especially for the ‘Remind rules’ action) had different emphases. Sometimes the teacher did not manage to have the robot saying exactly what she desired: “It’s often frustrating that I can’t pick exactly what the robot says, but at the same time, if I could there would definitely be too many buttons on the screen. The current layout is already difficult to get to grips with so there’s not much that can be done about this”. This variation of utterances led to slightly different actions grouped together and as such created some limit on the precision of control that the teacher had over the robot’s behaviour.

In summary, to allow the teacher to create a policy with a wide range of actions, there is a trade-off between the granularity of the actions available to the teacher (and consequently the precision of the policy) and the usability of the interface (additional actions/buttons might reduce the interface’s clarity and usability). When designing an application for SPARC, the level of actions (high versus low-level), the quantity of actions and the way the teacher can select them needs to be carefully considered.

Specify relevant features. As argued throughout this work, when teaching a robot, the human should also provide additional information to the learning algorithm to help it learn faster. Providing an interface between human-world features and ones making sense for the algorithm can be a challenge. In this study, the teacher’s interface replicated the game, and the teacher could see images of animals with some energy and the child moving them around, while the algorithm only had access to a vector of 210 dimensions. The teacher could select features of the environments relevant to the action and the action analyser would infer the dimensions of the state which were related to this list of features and use only these dimension for learning. This implementation proved successful in this experiment, but still required a sort of hand-coded human mapping and might be limited in its use (the designer had to create a mapping between human features and the ones used by the learning algorithm). Additionally, as mentioned in Section 6.6.2.2, the teacher mostly used simple feature selections as well as some elements that were not available to the algorithm. This might signify that this way of providing information to the learning algorithm is not ideal even for this type of interaction.

Other ways to help the robot learn by augmenting the demonstrations should be explored in future work. For example, when selecting an action, the teacher could specify the

rule they used and that the algorithm should replicate. Alternatively, the interface could more transparently display the state space to the teacher who would be able to see the evolution of the values and manually highlight the relevant dimensions.

React to and evaluate propositions. SPARC aims to reduce the workload on the teacher by having the robot proposing its actions to the teacher before executing them. This implies that the teacher needs a way to react to the propositions: preventing incorrect actions from being executed and ensuring correct actions are executed on time. To achieve this, first the proposed action needs to be displayed in a way that is quickly recognisable by the teacher. Then, the teacher needs to be able to prevent the action (through a button for example), enforce it or let be executed. Finally, the action needs to be rewarded by the human so the algorithm can learn. If the action is correct, its acceptance or execution could be enough for the learning algorithm. However, an action can be incorrect for many reasons, and specifying the reason might help the learning algorithm to learn a better policy faster. For example, in this study the ‘Skip’ and ‘Cancel’ buttons had different semantics for the learning algorithm (one informing that this specific action was wrong, while the other indicated the robot was not supposed to act for the time being). However, while having different semantics for the learning algorithm, these reactions appeared the same for the teacher: the robot did not execute the proposed action; consequently, the teacher used them interchangeably. Future work would benefit from designing better ways of feeding back why an action is wrong to the algorithm.

As mentioned previously, having a limited time for reacting to propositions also put pressure on the teacher to act quickly, potentially leading to errors or misuse of the interface. It might have been interesting to give control to the teacher over the correction window’s duration or the adaptive threshold limiting the suggestions. Providing the teacher with a way to mediate this time pressure (through additional control means on these parameters) might lead to a lower workload and potentially a better use of the interface.

Monitor the learner. When learning in the real world, the state of the learner might be hard to evaluate. General ML techniques for example have limited transparency: firstly, the state representation is probably not intuitive (such as a simple vector of 210 floating numbers between 0 and 1) and secondly, the learning process might be opaque,

closer to a black box than a transparent process. Furthermore, unlike simulation where a performance can be average over many runs, evaluating a policy for interacting with humans can take an extortionate amount of time. Consequently, teachers need to estimate online how accurate the robot's policy is.

In this study, this estimation could only be done through the robot's suggestions, the teacher did not have access to the robot's state or to the learning process. While being limited, this monitoring still provided more information than offline learning: the teacher was able to observe the learner's state through its propositions and potentially see the impact of their actions through the evolution of these propositions.

Future work could also adapt the interface. In this study, the choice for the teacher's interface was relatively straightforward: as the child was playing on screen, the teacher could simply have access to a duplicate of the game and interact on this representation of the world. However, this monitoring could also happen through other means. It could for example represent the state space in a compact and intuitive, yet complete, way to the teacher and present the policy through a set of rules learned from the supervision or simply use vocal commands to control the robot and verbal explanation of the robot's behaviour (Hayes & Shah, 2017).

Regardless of the interface's type, providing the teacher with a way to estimate to the learner's state is an important consideration when designing an application for SPARC. Providing the teacher with a transparent way to evaluate the robot's policy might help to create trust between the teacher and robot, potentially easing its autonomous deployment and increasing the chances that the autonomous robot will express a correct behaviour.

Recover from errors. A last functionality of the interface is recovering from errors. As mentioned previously, even a focused teacher will be prone to making errors, and the interface needs to provide a way to mitigate their effect. One type of error for this study was erroneous demonstration. The possibility of having incorrect instances in memory motivated the presence of the 'Remove' button, whose effect was to delete the closest instance of the last proposed action, thus preventing incorrect demonstration to impact future robot behaviours. However, the teacher did not have the opportunity to use it in the study, and as such its efficiency was not evaluated. Other types of errors happened, such as not correcting an action in time. However, in that case, no recovery

was provided. The interface could provide a way to have the robot apologise, thus reducing the impact of such incorrect actions on the trust between the child and the robot. In summary, as humans are not perfect teachers, the interface should allow them to compensate for errors and maintain both an efficient learning and a safe and useful interaction with the target of the application (such as children in this study).

6.6.3.3 HRI is Human Centred

A last consideration when applying SPARC to an HRI application is that HRI is human centred. When a robot is supervised in an interaction with a human, and especially with a child, the main goal of the teacher is to ensure that the experience for the child is optimal. Consequently, the teacher will be more focused on the child's behaviour than the robot's learning. For instance, if an action would help the child but hinder the learning for any reason (for example a child with special needs requiring a more proactive robot), the teacher would most certainly 'damage' the robot's learning to improve the child's experience. Except specific cases (for instance when using actors or informed participants), teaching a robot to interact with humans will always be a secondary activity or a by-product of the interaction.

Furthermore, when interacting sequentially with different users, the teachers will tailor their policy to the specific person involved in the interaction. Thus the human will not apply one homogeneous policy to all the interaction partners but potentially one per person. This is a challenge for ML as it further increases the complexity of the learning task. The algorithm has to either learn a much larger policy (covering all the different types of human partners) or learn a multitude of policies and be able to switch between them. Another method for reaching a personalised interaction could be to start with an initial general policy and then use SPARC with a teacher to refine and adapt it to the specific context of each interaction. That way, the robot would only have to learn to adapt its policy to each user, and the teacher might help to make this adaptation easier.

6.7 Summary

To conclude, this study demonstrated the applicability of SPARC to HRI. A robot was taught to interact efficiently with children in the real world. The teaching process led to a successful autonomous policy without perturbing the children (which would not have been possible with methods such as RL). In fact, the children even improved their performance in the game while the robot was learning to interact with them. This study

demonstrated that SPARC can be applied to real-world environment to teach robots to interact with humans, even if such environments are multimodal, with high dimensional action and state spaces, and highly sensitive to errors. Furthermore, SPARC provides stakeholders, such as teachers, a way to transfer their domain expertise to the robot while keeping control over the robot's behaviour. This progressive and transparent encoding of the human expertise (actions and relevant features of the state space) has the opportunity to create a stronger trust between the robot and its user, making an autonomous robot's deployment more acceptable and ethically safer.

Additionally, for the first time in this research SPARC has been applied to teach a robot to interact with humans in a complex and social environment. By using a novel algorithm adapted from Nearest Neighbours and designed to learn quickly in multidimensional states, the robot learned to produce a behaviour similar to the teacher's. Furthermore, this teaching was performed by a user who was not an expert in ML or robotics. While not leading to improvements in the children's learning gain as measured by the test, the behaviours from both the supervised and the autonomous robot led to improvements in the children's behaviours during the game.

This study provided partial support for the main thesis of this research: "Through receiving supervision from a human teacher, a robot can progressively and efficiently learn a policy for interacting with people, during the learning the robot will at all times display correct behaviour". Whilst the current implementation demonstrated limitations (for instance not reducing the teacher's workload over time), SPARC succeeded in its goal of allowing a user with limited knowledge of robotics to teach a robot to interact with humans in the real world in a multidimensional continuous social environment.

Chapter 7

Discussion

Chapter 2 highlighted the lack of robot controllers which provide adaptivity to a robot while at the same time offering a reduced workload for the people controlling the robot and ensuring that the robot's behaviour is timely and appropriate. Based on this observation, Chapter 3 presented Supervised Progressively Autonomous Robot Competencies (SPARC), an interactive teaching framework designed to allow robots to learn socially interactive behaviour by being supervised by humans. Then Chapters 4, 5 and 6 evaluated this approach in three studies, the last of which evaluated SPARC in real Human-Robot Interaction (HRI) consisting of a learning activity involving 75 children. Combined together, these chapters sought support for the thesis of this research:

Through receiving supervision from a human teacher, a robot can progressively and efficiently learn a policy for interacting with people, during the learning the robot will at all times display correct behaviour.

Chapter 4 presented a first a study comparing Wizard-of-Oz (WoZ) and SPARC. Results from this study demonstrated that a learning robot could reduce the human workload required to have it interact in the world without impacting its performance in the application interaction. This provide support for SPARC as a method allowing a robot to become progressively autonomous.

The second study presented in Chapter 5 explored how the control provided to the teacher by SPARC impacted the teaching of an efficient policy. This study showed that by giving the human teacher control over the robot's actions, SPARC could ensure that the executed robot behaviour fits the teacher's desires which led to a faster, safer and easier teaching.

As the first and second studies were focused on the relation between the teacher and the robot, they had to use a repeatable and controlled environment to study SPARC. In contrast, the last study applied SPARC to a real-world HRI: child tutoring. In that

study, an adult had to teach a robot to tutor children in an educational game. Results demonstrated that after having been taught using SPARC in multidimensional and generic state and action spaces, an autonomous robot displayed an efficient social policy suited to interacting with children. The policies from the supervised and the autonomous robots presented similarities, and both policies resulted in more positive children behaviours compared to a passive robot. Whilst not reducing the workload on the human during the teaching process, SPARC demonstrated its applicability to complex, multimodal and high-stakes environments. During the teaching process, the teacher could ensure a useful robot policy, which was maintained even after the teacher exited the control loop.

This chapter provides an overarching perspective on the results presented in earlier chapters. It presents how the three studies in this work answer the research questions posed in Chapter 1. Then, Section 7.2 presents the limitations of the approach proposed in this work, SPARC. Section 7.3 discusses the more general impacts this research may have and the ethical questions raised by teaching robots to interact with humans. Finally, the last section proposes axes where SPARC could be extended, to on the one hand learn more about how people can teach robots, and on the other hand improve SPARC's usability and application to HRI and other fields.

7.1 Research Questions

This section will revisit the research questions identified in Section 1.2 and explain how the work presented in this thesis addressed them.

RQ1 What are the requirements of a robot controller for social HRI? Based on a review of the different fields of application of social HRI, we defined three requirements a robot controller should meet to ensure an efficient interaction. First and foremost, the robot's behaviour needs to be constantly appropriate: as robots often interact with vulnerable populations, their behaviour needs to be safe for the people they interact with and appropriate to achieve a desired interaction outcome. Secondly, the robot should be adaptive, i.e. it should be able to generalise to unexpected situations, but should also personalise its behaviour to the different users it interacts with, and be able to learn, improving and extending its policy. Thirdly, the robot needs to be as autonomous as possible, or at least impose a low workload on its supervisor.

This implies that the robot needs to find ways to learn about its environment and reach autonomy without relying on random exploration as this would violate the first principle. We think the robotic community, and especially HRI, should strive toward more autonomous robots, and could take a stronger inspiration from Interactive Machine Learning (IML) as it shows strong promises for teaching robots and could enable them to learn complex tasks such as interacting with people.

RQ2 What interaction framework would allow a human to teach a robot while meeting the requirements from RQ1? To meet the three requirements expressed as answer to RQ1, we proposed SPARC, a new teaching framework for robots which provides control over the robot's actions to a teacher and use this control to learn in a safe way, validating the first and second requirements. Secondly, by allowing the teacher to passively accept the robot's propositions, we aim to decrease the workload on the teacher over time and progressively provide the robot with autonomy. By taking inspiration from IML, SPARC aims to fill a void in the HRI research: online learning for interaction with humans.

RQ3 Could a robot decrease its supervisor's workload by proposing actions based on observing the supervisor's earlier decisions? Study 1 showed that providing a supervised robot with learning can reduce the workload on its supervisor. Furthermore, study 2 demonstrated that, compared to the literature, SPARC is an efficient way to enable safe teaching and requires a comparatively low workload. Similarly to methods from Learning from Demonstration (LfD) (Liu et al., 2014; Sequeira et al., 2016), SPARC provides an alternative to WoZ and opens new opportunities to have robots learn social behaviours from observing humans control. However, unlike classic LfD methods, SPARC could produce results supporting the teacher during the learning process, informing them about the robot's knowledge and potentially creating trust between the robot and its teacher, qualities still lacking in most other learning frameworks.

RQ4 When teaching a robot, how does the control over the robot's actions by the teacher impact on the teaching process in terms of performance, effort and teaching time? Results from study 1 and 2 indicated that by informing the teacher in advance of its actions, the robot ensures that its final behaviour is vetted by the teacher. This implies that even in early phases of the learning, when

the robot behaviour is not adequate yet, the teacher can prevent the robot's lack of knowledge to negatively impact the world. Furthermore, this control helps the teacher to steer the robot toward useful parts of the environment and to demonstrate a better policy, making the teaching faster, safer, more efficient and lighter (as requiring a lower workload) compared to methods which do not use human supervision while learning. This finding is especially relevant to IML, as often human teachers are offered limited control over the robot's policy (Thomaz & Breazeal, 2008; Knox & Stone, 2009). While being a challenge, providing the teacher with this control can have significant positive results on the learning progress.

Research questions 5 and 6 apply to the special case when SPARC is used to teach a robot to interact with people. As such, the resulting interaction is as proposed in Figure 3.3: a triadic interaction between a human-target, a robot and a human teacher.

RQ5 What impact does SPARC have on the performance of the child-robot interaction, the comfort for the teacher and the robot's learning? When being used to teach a robot to interact with a person, SPARC does allow the robot to display an effective behaviour (as demonstrated by the improvement of children's engagement with the educational task in the supervised condition in Chapter 6). However, this performance in the application interaction might come at a cost for the teaching interaction. As the teacher needs to monitor the human in the application interaction, they also have to react to the robot's suggestions, this dual task might lead to a heavier workload than classic tele-operation methods. This is one of the drawbacks of SPARC compared to methods based on LfD to gather information, but as mentioned earlier, ways exist to mitigate this issue and allow this interaction between the teacher and the robot to lead to positive results for the teacher.

RQ6 After having been taught using SPARC, could a robot behave autonomously in a social context? In Chapter 6, we used SPARC to teach the robot and then deployed the robot to interact autonomously. During this autonomous interaction, the robot applied a policy similar to the demonstrated one, and the impact on the children's behaviour was close to the one in the supervised condition. Consequently, in this study, the robot managed to behave socially in an autonomous

fashion in a complex and multimodal environment after having been supervised by a person in a learning phase. This finding is one of the most important of the thesis as it demonstrates the potential of SPARC to teach robots complex social autonomous behaviours.

7.2 Limitations of SPARC

This section discusses the limitation of SPARC, specifically to the range of domains it could be applied to.

7.2.1 Requirement of a Human in the Loop

The first of these limitations is the potential requirement for a human to supervise the robot, even after it has learned a policy. As stated earlier, SPARC aims to move away from WoZ or other tele-operation methods by learning an efficient policy from initial human supervision. In its original framing, SPARC did not aim to create a fully autonomous agent acting without supervision, but instead gradually learned from monitoring commands given to the robot, taking over from the supervisor as the interaction progresses. As such, the learning is not split into two distinct exploration and exploitation phases, but is a step in which the robot transitions smoothly from having no knowledge to having incorporated knowledge from observing the teacher's behaviour. Because of this, the workload on the supervisor would decrease as the robot learns, while a high performance in the domain application would be maintained due to the oversight provided to the teacher. However, Supervised Autonomy still requires a human in the loop and as such presents limited applicability to fields where robots are expected to be fully autonomous, for example to substitute human labour.

Despite not being the original goal, SPARC can still be used to create a fully autonomous robot (as demonstrated in Chapter 6). In this case, the training process would be similar to LfD, with a training phase using SPARC and then the exploitation/deployment phase of the autonomous behaviour. However, with SPARC, differences remain. First, during the training phase, instead of passively receiving commands from the teacher, the robot would proactively make suggestions to the teacher. As presented in Section 6.6.1.3 this aims to reduce the workload on the teacher during the training phase, provide more datapoints for the learning and inform the teacher about the state of robot's knowledge, potentially creating trust between the robot and its teacher. Secondly, even after being

deployed to interact autonomously, the teacher could still take back control using SPARC, thereby refining the policy. Alternatively, if a different behaviour has to be applied (e.g. if the robot interacts with a child with special needs rather than a typical one), the teacher can take over using Supervised Autonomy to ensure a personalised experience for this specific interaction.

7.2.2 Reliance on Human's Attention

A second limitation of SPARC lies in the constant need for the human's attention and the presupposition that human teachers will always ensure appropriate robot behaviour if given the opportunity. Throughout this research, we assumed that even if the robot behaviour may be mostly correct, the supervisor would be attentive to the robot suggestions and ready to correct any error at any time. This assumption is similar to autonomous cars using a safety driver or Tesla Autopilot requiring continuous human supervision¹. The agent is fully autonomous but may make mistakes and as such, a person needs to be ready to correct these errors before they impact the world. However, as demonstrated by the accidents in 2017 and early 2018 involving these supervised autonomous vehicles, this assumption is often violated, and a short moment of inattention may have dire consequences². By observing a seemingly correct agent behaviour for an extended period of time, the human supervisor might start to overtrust the agent, missing the occasion to react in time to anticipable errors potentially leading to frustration or death in the case of autonomous vehicles². Nevertheless, some ways exist to mitigate this limitation but have not been applied to autonomous driving or general IML. For example, with Supervised Autonomy the agent in advance informs the supervisor about its actions, similarly a car could display the planned trajectory on a screen or in augmented reality. This would provide the supervisor with more time to analyse the situation potentially allowing them to react in a timely fashion. Alternatively, the agent could communicate a lack of confidence in its actions or its interpretation of the environment, informing the supervisor that attention is specially required in that moment.

¹"Every driver is responsible for remaining alert and active when using Autopilot, and must be prepared to take action at any time." https://www.tesla.com/en_GB/autopilot

²Cf. the Tempe and Mountain view accidents reported by the media in early 2018.

7.2.3 Time Pressure

As pointed out in Section 3.5.1, with SPARC, the presence of the correction window and the auto-execution of actions may lead to issues. The length of the correction window is a design decision and depends of the application, to be applicable and make use of the auto-execution of actions as a way to reduce workload, the validity window of an action needs to be wider than its correction window. That way, actions approved passively are still valid when executed. To increase the application of SPARC to a wider range of situations, the correction window has to be as narrow as possible. In contrast, the supervisor needs a correction window as wide as possible, to provide them with enough time to process the action and cancel it if required. Correction windows too narrow would put additional pressure on the supervisor to react in time or even prevent them to avert undesired actions. This results in two effects having opposite requirements on the correction window. However, while a longer correction window would only limit SPARC's applicability in some situations, one too short could have negative consequences. As such, this need of a correction window wide enough to allow the teacher to react is probably one of the main limits of SPARC as it produces a significant delay in the robot's actions and reduces the range of domains SPARC can be applied to.

However, this requirement of a correction window wide enough for the teacher can be mitigated in multiple ways. For example, instead of communicating the action the robot is directly about to do, the robot could communicate a plan with multiple steps announced in advance. That way, the teacher would be informed beforehand of the next few steps and could anticipate their impact and react to any future actions instead of limiting their evaluation to the next one. Alternatively, the robot could adapt the length of the correction window to its confidence in the proposed action; for instance, an action with a low confidence would be given more time to be corrected. Likewise, each type of action could have a dedicated value for the correction window, for example actions needing to be executed quickly (such as emergency breaking for example) would have a much shorter correction window than other actions with less time constraints. Finally, a last possibility could be to allow the teacher to manually select the duration of this correction window, consequently letting them be in control of the pace of the interaction. In that case, the teacher might prefer to start with a long correction window and make a limited use of the auto-execution in the early phases of the interaction to be able to focus

on the teaching process; but in later stage of the interaction, they might reduce this correction window to profit from the auto-execution of actions and reduce their workload.

7.2.4 Overloading the Teacher

By giving an active role to the robot in the teaching process (through proposing actions), SPARC requires the teacher to simultaneously monitor two autonomous agents: the target user and the robot. This requirement might lead to another risk with SPARC: overloading the teacher, rather than reducing his workload. If the teacher has to correct more actions than they would have selected, their workload would not be reduced but increased. While still providing useful information for the learning algorithm (and more than only the actions selected by the supervisor), this supplementary workload on the teacher is not desired. As explained in Chapter 6, this could lead to erroneous teacher's behaviours hindering the learning and potentially increasing the risks in the interaction. For example at some points in the study presented in Chapter 6, the teacher just cancelled actions as soon as they arrived, even before she had time to evaluate them. While reducing the workload on the teacher by not requiring them to evaluate the proposed action, this behaviour might limit the efficiency of the learning algorithm by giving it incorrectly labelled datapoints.

Overloading the teacher is a serious issue and might have negative consequences both for the robot's learning and for the experience of the user involved in the application interaction. As such, mechanisms must be present to ensure that this does not happen. The learning algorithm needs to have the right balance between suggestions and waiting periods to allow the teacher to assess and provide a correct evaluation of the proposed actions if needed. Alternatively, the teacher could be provided with a direct way to impact on the rate of suggestions and on the time before the auto-execution of actions. This would give them control over their workload and might allow them to teach the robot more efficiently.

7.2.5 Interface

The interface between the teacher and the robot is key when applying SPARC or other IML methods. Simple interfaces can be easy to create and are easy to use by the teachers, however they might only have limited efficiency in the learning process. For instance, approaches using only feedback (i.e. numeric evaluation) require a single one-way communication channel between the teacher and the robot, and this channel

only needs to send a scalar evaluation of the agent's actions. Consequently, both the design of the interface for the teacher and the communication are simple but their efficiency is limited. On the other hand, to provide full control and accountability over the robot's actions, SPARC requires two-way communication. Firstly, the teacher needs to receive input from the robot, such as its intentions, to decide if the action is valid or not. Secondly, the teacher needs to send information to the robot: feedback about the intentions, cancelling and correcting actions if required, but also demonstrating by selecting actions for the robot to execute. As the robot may have access to hundreds of actions, assigning one button per action is not feasible, other ways of controlling the robot need to be found. Furthermore, as mentioned in Chapter 3, with SPARC the teacher can also provide additional information to the algorithm to speed up the learning. In summary, the interface between the robot and the teacher needs to provide the teacher with the robot's intentions, and allow the teacher to preempt proposed actions, select any action and provide additional information to the learning algorithm. An interface providing all these features can easily be overwhelming and difficult to use by naive users, increasing even more the required workload to control and teach the robot. As such, for applying SPARC to complex environments, an investment will need to be made in the interface to make it intuitive and clear.

For instance, in Chapter 6, the Graphical User Interface (GUI) used by the teacher represents the current state of the game, with some buttons for accessing a subpart of the action space, but the majority of actions were inferred by the way the teacher moved items on the screen and how they selected items. An alternative way could be to use natural language. Natural language is an intuitive channel for humans and its open-endedness makes it suited for applications of SPARC where the teacher can speak and where the time required to vocalise commands is not critical, such as a robot assistant at home.

7.3 Impact

7.3.1 Pushing the State of the Art

At the time of writing, few other methods have been applied to teach robots to interact with humans. As mentioned in Chapter 2, the main other approaches were Learning from the Wizard (LfW) and LfD, they used demonstrations from a WoZ setup or from interactions between humans, and from these demonstrations learned a policy

offline (Knox et al., 2014; Liu et al., 2014; Sequeira et al., 2016). Due to the challenges when learning during the interaction (high stakes of actions, limited datapoints and complexity to maintain a policy providing useful data), online learning and IML were seldomly used to teach robot to interact with humans, these approaches were mostly focused on teaching agents to interact in other non-social environments. Additionally, most of the previous methods in IML only provide limited control to the human teachers, reducing them to “feedback providers” while humans could and should provide much more information to learning agents (Amershi et al., 2014)

By proposing SPARC, we wanted to push IML to give more power to the teacher, by making better use of people’s ability to steer the learning process, and by making it safer and more comfortable for the teachers. By keeping a human in control of the robot’s actions, SPARC enables robots to learn online in sensitive environments. Furthermore, with the combination of proposition, correction and selection of actions, SPARC has the opportunity to reduce the workload on the teacher. Hence, this approach is fit to control and teach robots to interact with humans as it follows the requirements presented in Section 2.1.2. With this method, we successfully demonstrated in Chapter 6 that robots can be taught online to interact efficiently with humans, while ensuring a constantly appropriate policy. By using a new algorithm and keeping the teacher in control, SPARC allowed a robot to learn a social and technical policy in a high dimensional and multimodal sensitive environment.

By demonstrating its applicability to teach robots to interact with humans from *in situ* supervision, SPARC pushes the state of the art both in HRI and IML. By building on LfD, SPARC keeps its advantages: allowing to teach behaviours complex to define or not known in advance. Furthermore, by including an online learning component and keeping the teacher in control, this new method might allow robots to be deployed in new contexts where they are absent today.

7.3.2 Empowering Non-Experts in Robotics

Other methods used to teach robots to interact with people use post processing on previously gathered demonstrations to learn a policy offline. While leading to positive results, these approaches require significant engineering after the demonstrations have been recorded to learn a robot behaviour, and provide limited opportunities to refine the behaviour after the learning is over (Liu et al., 2014; Sequeira et al.,

2016). This implies that they cannot be used solely by end-users not knowledgeable in machine learning, experts in robotics have to interpret the data provided by the domain expert to design the behaviour. In contrast, by mixing together the collection of data and the learning, SPARC removes this barrier between data collection and use. That way, end-users can themselves directly teach a robot to interact as they desire. SPARC provides an opportunity for anyone to personalise their robot without requiring technical skills. As demonstrated in Chapter 4 and 5, from a single algorithm and state representation, SPARC can lead to different behaviours adapted to the teacher's strategy and preferences. Combined with efficient interfaces and learning algorithms, approaches such as SPARC have the potential to democratise robotics by allowing anyone to teach a robot to interact efficiently in a wide range of domains. Robot developers and designers could use this learning ability to deploy robots as *blank slates*, with just a way to perceive the world, act on it and interact with a teacher, and let their behaviour be defined by their users. These users would start filling the blanks, creating their own robot behaviour, teaching their robot how to fulfil their personal needs. As defended by Fails & Olsen Jr (2003) and Amershi et al. (2014), by allowing end-users to teach an agent to behave as they desire, IML methods have the potential to ease the deployments of technology and reach new application domains faster. This might allow users currently excluded from using robots (due to lack of interest from developers and lack of technical skills from the users) to profit from this new technology.

While providing many opportunities, deploying a blank robot able to learn complex policies would require significant engineering pre-deployment to have a wide enough state and action space, and a learning algorithm and interface efficient enough to reach useful policies. However, as demonstrated by the study in Chapter 6, this is achievable. A robot can be deployed with large state and action spaces and then using algorithms designed to learn quickly from teachers in complex environment, a non-technical person can teach a robot a complex social policy. Furthermore, by providing additional tools to the users to widen or refine the state and action space, this teaching could be applied to even more applications.

7.3.3 Robots and Proactivity

Another way to interpret SPARC is as a way to provide anticipatory powers to a robot. By using Machine Learning (ML) and proposing to execute actions, the robot is actually

taking the initiative to take an action without executing it straight-away. For instance, a proactive robot assistant would anticipate its user's needs and desires and would propose help or services without having to be asked (Mason & Lopes, 2011). This capability has two applications: first it allows robot users not to have to ask supportive behaviour from the robot every time they need it, and second, it means that the robot could even support its user when they do not realise help would be useful.

By having the ability to learn new actions, or what action it should do, such a robot would move from a simple tool to an adaptive partner able to support its user in a large number of tasks. Finally by informing the nearby people of its actions, such a robot assistant would only execute actions deemed useful by its users, limiting the chance of negative outcomes.

7.4 Future Work

The work conducted in this thesis explored how robots could be taught to interact with humans and proposed a novel interaction framework, SPARC, to enable such a learning. However, SPARC could be extended in many ways and its principles applied to other applications.

7.4.1 Application Domains

SPARC emerged from the DREAM project, a European project aiming to develop new Robot Enhanced Therapies (Thill et al., 2012; Esteban et al., 2017). Due to the limitations when working with children with Autism Spectrum Disorder (ASD), SPARC has only been applied to HRI in the context of tutor robots, to teach a robot to support child learning. Nevertheless, the principles underlying SPARC could be applied to a much wider range of applications in HRI and in other fields (cf. Section 7.4.2). For instance, SPARC could be applied to teach robots a therapeutic behaviour for Robot Assisted Therapy (RAT) while keeping the therapist in control of the interaction (as aimed by DREAM). By learning from the therapists, robots could be applied more easily in different therapeutic scenario without requiring a workload as high as WoZ.

SPARC would also show promises in numerous other applications in social HRI: from assistant robots at home to collaborative robotics, including robots in hospitality, the military or industry. For example a robot could learn the preferences of a user and act as an embodied personal assistant, connected to devices in the house, calendar

on the internet and supporting its users in routine tasks. Such a robot could learn to anticipate its user's needs and propose to provide support proactively. In Human-Robot Collaboration (HRC), similarly to the work presented in Munzer et al. (2017), a robot could learn its partner's preferences, informing them of its actions and helping them to complete the task faster and easier. As such, future work should apply SPARC to other use cases of HRI and explore how it could be used to teach agents to interact in other complex or high-stakes environments in HRI or outside.

7.4.2 Learning Beyond Imitation

Another potential feature of SPARC, and other methods based on demonstrations, not evaluated in this work is reaching capabilities beyond the demonstrations. As SPARC uses demonstrations and corrections from a human teacher to learn, by applying Supervised Learning (SL), the optimal outcome would be to match the teacher's performance. However, if the algorithm possesses or learns a value function or the teacher's goal, instead of reproducing the teacher's policy, the agent could improve its policy around the demonstrated one and potentially become better than the teachers themselves. This achievement have been accomplished by Abbeel & Ng (2004), by using Inverse Reinforcement Learning.

Alternatively, instead of having the robot reaching capabilities beyond human ones on its own, a human-robot team could also together reach this kind of performance. For examples, Kasparov (2010) proposed "Advance Chess", a new type of chess were players have access to a computer to help them during their decisions. This combination of human and machine, aims at profiting from the best of both worlds and would allow humans to make better use of their intuition and creativity while using computer's certainty to save human calculations. Similarly, a learning agent could interact with the human in a mixed initiative framework, such as SPARC, where the agent could suggest actions (such as moves in a Go or Chess game) and the human could accept them or refuse them. That way the human would be in control of the interaction, preventing potential errors from the artificial agent to have negative impact, while still being open to opportunities they did not anticipate. Hence, the team could reach together super-human capabilities while ensuring, with the presence of the human in control of the interaction, that the behaviour would be at least of human performance. Additionally,

this mixed initiative interaction might provide the human with the opportunity to learn from the robot, if the robot proposes better than anticipated actions.

However, to reach these super-human behaviours, the agent requires a way to learn in addition to the human. The agent needs to have access to a second level of learning, for example through receiving rewards directly from the environment or through learning a reward function from the human demonstrations (such as with Inverse Reinforcement Learning). For example, the human could provide a mixture of demonstrations, rewards and high-level goals which could be used to learn such a reward function or to update a planner with more correct models allowing the robot to reach the goals faster. Alternatively, the robot could learn simultaneously from the human and in simulation. The human would use SPARC to stay in control of the interaction in the real world; and the environmental and human feedback from these real interactions could help refine the simulation. That way, in the real world, the robot's behaviour would stay appropriate, as a human would be in control of it, while the robot would have freedom to explore in simulation.

7.4.3 Sustained Learning

This work, and most of the general research in robotics, considers the problem of learning single tasks in isolation. However, once deployed, robots need to address the challenges of lifelong learning (Thrun & Mitchell, 1995), being able to continuously learn in different domains and transfer knowledge from one situation to another. When interacting in the real-world for extended periods of time, robots cannot rely on offline sessions with engineers to improve their behaviour. By using online learning, methods such as SPARC could provide this online refinement of behaviour and potentially help the robot to know which parts of the policy are transferable. Furthermore, by allowing the end users to provide commands and information about the desired policy at runtime, SPARC reduces the need of computing experts once the robot is deployed.

SPARC also assumes that the teacher is constantly supervising the robot, however, when deployed in the real world, the teacher might have to leave the robot interact unsupervised for limited periods of time, and the robot needs to adapt its policy to this change. Work has been done in that direction (Faulkner et al., 2018) and could be combined with SPARC.

Another challenge of learning over long periods of time is the policy's dependency in time. For example, in Chapter 6, the temporal aspects of the interaction were taken into account only by including some notions of time since events in the state definition. The robot was not doing any temporal planning or explicitly taking time into account when behaving. However, to sustain continuous long term interaction, spanning multiple hours or days, the dependency in time of the policy has to be taken into account through other means. One way is to learn spatio-temporal features relevant to the human's behaviours, expectations and desires(cf. STRANDS project; Hawes et al. 2017 and Soh & Demiris 2015). For instance, a robot assistant at home should know its users are typically going to work every morning, except weekends and holidays, and a human teacher could help the robot to interpret these elements related to time. Finally, to sustain learning over long periods of time, robots also need algorithms that can scale well with a large number of data.

7.4.4 Interface With the Teacher

Another axis to improve SPARC, and which is critical to consider when tackling new applications, is the interface with the teacher. One of the main limitation of SPARC is also what provides its strength: the inclusion of a human in the action selection process. Including this person and giving them the opportunity to preempt and select any actions comes with limitations on the interaction. The robot needs to communicate its intentions and the human needs enough time to correct them before they impact negatively the environment or other humans interacting with the robot and the teacher needs to inform which actions the robot should execute.

Consequently, the interface used by the teacher to control the robot is key and should mitigate these limitations. Further work should explore how to provide the best communication between the teacher and the robot. For example, in the case of a GUI on a tablet or a phone, the interface could combine buttons and a representation of the world where the robot could describe how it plans to act or its expected trajectories. Similarly, the teacher could use this representation of the world to select actions for the robot to execute (as used in Chapter 6 but also including long-term information). Designing these GUI for SPARC could use the knowledge obtained from designing application for phone for example (Joorabchi et al., 2013).

However, GUI might not be the optimal way to communicate with the robot, as they might not scale well with a high number of actions and might require additional hardware. Future work should explore alternatives interfaces, such as natural language, to control a robot through SPARC. This would raise many challenges, such as natural language understanding, or creating verbal commands describing the robot's actions, intentions or explanation clearly yet concisely (Hayes & Shah, 2017). Despite these challenges, language possesses the qualities required to communicate between a teacher and a robot: familiarity for humans, open-endedness of description and precision for example. Another area of research which would improve SPARC is Brain Computer Interfaces. For example, when witnessing an error, the brain creates a specific pattern which can be detected using EEG (Gehring et al., 1993), that way instead of an explicit correction window, such an interface could automatically detect errors in robot suggestions without requiring the teacher to explicitly cancel the action. By having a much smaller delay between the proposition and its execution, SPARC could be applied to more applications.

7.4.5 Algorithms

In the future, SPARC should be combined with richer learning algorithms. The three examples provided in this thesis represent only a small part of the algorithms SPARC can be used with. More advanced ML have the potential to allow SPARC to learn faster and more efficiently; and, as mentioned previously, potentially reach super-human performance in the task. However, as mentioned in Chapters 2 and 3, many challenges remain when using learning for HRI. The first one, the main one tackled by SPARC, is the high stakes of the interaction: errors might lead to disastrous consequences. In addition, the data efficiency is fundamental: when interacting with people, collecting information about human's reaction can be costly or take a large amount of time. As such, each datapoint should be used with high efficiency and the human included in the loop should provide additional knowledge to deal with this scarcity of data. Alternatively, the algorithm could combine data accumulated from different teachers to learn a more general policy. However, this could lead to a transfer problem, assumptions and policies correct with one user might not be valid anymore with another one. Doing this generalisation might decrease the potential for personalisation that ML provide. One way to address this personalisation vs generalisation issue is to group people by similarity and learn to detect the group of a person and a policy adapted to this group to

reach a better policy (Brunskill & Li, 2014). Alternatively, the robot could learn or already possess a general policy and then use SPARC to refine it and adapt it to its user.

SPARC and humans in general could also be used to teach hierarchical strategies (Barto & Mahadevan, 2003). With hierarchical learning, agents learn subpolicies used to complete subgoals, and then combine these subpolicies to reach higher goals. By helping an agent to create subpolicies and informing it which ones are relevant to specific context, a human could allow an agent to learn to solve complex tasks much quicker. This teaching on multiple levels presents a challenge for SPARC, as in the current implementation, it only considers actions one by one and the teacher cannot inform about goals. This would require a way for the teacher to create higher level actions, organise them and switch between them. However, using a human provides a strong potential to quicken the learning of complex and rich policies.

Additionally, instead of providing demonstrations of a policy to follow, the teacher could also give symbolic rules defining the robot's behaviour. This alternative way of teaching could generalise faster than simple demonstrations and might allow teachers to define complex behaviours easily and without having to encounter each situation to show the robot how to behave.

Another challenge for algorithms used with people is the fact that people are not static entities. As mentioned in Section 2.3.4, different people will use different teaching strategies. Furthermore, as seen in Chapters 4, 5 and 6, in Thomaz & Breazeal (2008) and MacGlashan et al. (2017), human teachers adapt their teaching strategy overtime. Human policies and feedback are moving targets, and algorithms used to learn from people need to take into account these variations and evolutions of behaviours.

7.5 Summary

This chapter started by revisiting the research questions identified in Section 1.2, describing how this work addresses them. We then presented the main limitations identified for SPARC (requirement of a correction window and attention from the teacher, potential increase of workload on the teacher and complexity of the interface). We presented potential ways to address these limitations when designing interactions involving human teachers. We continued with discussing the potential impact of SPARC on social HRI and presenting how it pushes the state of the art further. Finally, we

7.5. SUMMARY

proposed directions to extend the research about SPARC, exploring how to increase its range of application and its efficiency.

Chapter 8

Contribution and Conclusion

8.1 Summary

The main thesis defended in this work is that:

Through receiving supervision from a human teacher, a robot can progressively and efficiently learn a policy for interacting with people, during the learning the robot will at all times display correct behaviour.

To explore this thesis and the research questions arising from it, Chapter 2 reviewed the application field of social Human-Robot Interaction (HRI). From this overview, we drew three requirements for a robot to interact efficiently in human environments. The robot needs (1) to constantly have an appropriate policy, (2) to be adaptive and (3) to require a low number of inputs from humans. Based on these requirements, we analysed the current controllers for robots in HRI and noted a need for a robot controller which met these requirements. However, we highlighted that, while not having been applied much to HRI yet, Interactive Machine Learning (IML) does hold considerable promises for providing a robot with social interactive behaviour.

In Chapter 3 we addressed this opportunity by proposing a new approach, the Supervised Progressively Autonomous Robot Competencies (SPARC), to teach robots interactive capabilities and which would as such be applicable to HRI. SPARC aims to provide control over the robot's actions to a human and uses this supervision to learn a policy online. To achieve this goal, SPARC is based on a set of principles allowing this teaching interaction to be efficient: the teacher can select actions for the robot, the robot can propose actions to the teacher, the teacher can preempt robot's propositions before their automatic execution and, finally, the robot learns from the teacher's input and feedback to improve its policy. These principles ensure that the robot's executed policy is constantly appropriate, while reducing the human workload over time, leading to an autonomous behaviour if desired.

Once the method and the principles defining the interaction between the human teacher and the robot were set, we evaluated this approach in three studies with increasing ambition. Before testing SPARC in a real world interaction with people (in Chapter 6), we needed to evaluate if the principles underlying SPARC could reduce the workload on the teacher and how the control over the robot's actions impacts the teaching process. A key effect to evaluate was if this control was sufficient to ensure a constant appropriate robot behaviour. The two first studies (Chapters 4 and 5) included people only as teachers, but not as the target of the taught policy. This allowed us to gather initial information on SPARC in controlled and repeatable environments without having to tackle the challenges of interacting with people in the real world.

The first study, presented in Chapter 4, evaluated the interaction between the supervisor and the robot in a controlled and repeatable environment inspired by Robot Assisted Therapy (RAT), where the patient interacting with the therapeutical robot was replaced by a second robot simulating a child. The study showed that SPARC could allow a robot to learn, subsequently reducing the teacher's workload by decreasing the number of actions required from the teacher to control the robot; and this decrease of human workload did not impact negatively the performance in the interaction. In summary, by learning the robot could maintain a high performance in the interaction while requiring lower efforts from the human supervisor.

Once SPARC was demonstrated to allow a robot to learn and decrease the workload on its supervisor, we evaluated in Chapter 5 how the control provided by SPARC to the teacher could improve the learning process. To explore this aspect, we designed a second study comparing SPARC to Interactive Reinforcement Learning (IRL), an alternative approach used to teach robots, but providing the teacher with little control over the robot's actions. The results showed that the additional control provided to the teacher by SPARC helped them guide the robot to relevant parts of the environment, thus improving the teaching process and making it safer and easier.

Since the two first studies demonstrated the efficiency of SPARC to teach a robot to interact in virtual environments and ensured that the control provided by SPARC could guarantee the appropriateness of the robot's actions, SPARC was ready to be evaluated in a real human-robot interaction scenario. We decided to select the context of tutoring children in the wild as this classic application of HRI encompasses many challenges

faced by robots interacting with humans. This last study applied SPARC to teach a robot a social and technical policy to support child learning. This study had three main goals. First, demonstrating the applicability of SPARC in teaching robots to interact socially with humans, consequently addressing the main thesis of this research. Secondly, exploring the impact of SPARC during the teaching phase on the two people involved in the triadic interaction. And finally, evaluating if, after having been taught by a person using SPARC, the robot could interact successfully with children in an autonomous manner. Results from this study demonstrated that during the teaching phase, the robot's behaviour was efficient and appropriate to the interaction, but might have required a relatively high workload from the teacher. And finally, when deployed autonomously, the robot could interact efficiently with children, leading to a policy selecting actions similar to those selected by the teacher and having similar effects on the children's behaviours. Consequently, SPARC was shown to enable the teaching of rich social behaviours in a multimodal and multidimensional environment where incorrect actions can have important consequences.

8.2 Contributions

As mentioned in the introduction, the main scientific contributions of this thesis are as follows:

- **Design of a new interaction framework for teaching agents in a safe manner.** One of the main contribution of this research is SPARC: a new interaction framework enabling robots to learn to interact with people from *in situ* human supervision. This method is the cornerstone of this thesis and aims to allow robots to be adaptive, while constantly ensuring an appropriate robot policy and requiring a low workload from humans.
- **Evaluation of SPARC in three studies.** SPARC has been tested in three studies, which were specifically developed to explore different aspects of the human-robot interaction involved in SPARC and culminated with the teaching of a robot to interact with children in the wild.
- **Demonstration that by learning from and proposing actions to a human supervisor, a robot can reduce their workload without reducing the performance.** As demonstrated in Study 1, by progressively improving its suggestions

based on previous human decisions, the robot can reduce its supervisor's workload.

- **Demonstration of the importance of control over the robot's action when teaching a robot to interact.** As demonstrated by the second study, giving the teacher control over the robot's actions has consequent impacts: it makes the teaching process safer, quicker and more efficient and it also provides the opportunity to reduce the workload on the teacher.
- **Design of a lightweight algorithm to quickly learn from demonstration in complex environments.** Learning to interact in complex environments, where states and actions are defined on a large number of dimensions is a challenging task, especially when the number of datapoints accessible to learn is low. By using a selective reduction of dimensions decided by the teacher, this new algorithm inspired by nearest neighbours allowed a robot to learn to interact in the complex environment, i.e. child tutoring with few datapoints.
- **Application of IML to safely teach robots social autonomy from *in situ* human supervision.** Finally, the second main contribution of this research is a demonstration of online learning of an interactive policy for HRI supported by human supervision. Through SPARC a robot learned a policy from scratch by interacting with people and applied it successfully in further autonomous interactions. The complexity of the world in which the robot interacted should be noted: the robot needed both technical and social knowledge about the interaction and had to behave appropriately at each step of the learning process. Furthermore, both the action and state spaces were large and multimodal. Despite these challenges, only 25 short demonstration sessions (consisting of four times 1.9 minutes each in average) were required to reach an efficient policy (providing gains between 10 and 30% on relevant interaction metrics – learning items visited and points achieved in the game for example). This has real world implications as it shows that by using SPARC, an agent can learn an effective social policy from a generic perception of the world and a large set of actions.

8.3 Conclusion

The thesis presented here is: "Through receiving supervision from a human teacher, a robot can progressively and efficiently learn a policy for interacting with people, during

the learning the robot will at all times display correct behaviour". To support this thesis, I propose Supervised Progressively Autonomous Robot Competencies (SPARC), a new machine learning framework for artificial agents seeking to give the teacher full control over the agent's actions. The agent learns from supervision in a safe and efficient way and progressively becomes autonomous. By proposing and evaluating SPARC in three studies, this work increases our knowledge in both HRI and IML, especially on how robots could be taught to interact in complex and sensitive environments, such as therapy or education. We showed that, while being a challenging task, providing a robot teacher with control over the robot's actions leads to significant advantages: the learning can be fast, efficient and lightweight for the teacher. And most importantly, by preventing the robot's errors in the early stages of the interaction, robots can be taught to interact well in high-stakes environments.

Finally, by showing its efficacy in child tutoring, I confirm the applicability of SPARC in complex and sensitive environments such as HRI. This demonstrates the potential for SPARC to provide non-experts in computing the ability to teach robots rich behaviours in large and complex environments and shows potential for deployment in more interactive domains.

Glossary

Correction window The correction window is the time provided to a supervisor to correct an action before its execution. 129

Validity window The validity window is the duration of validity of an action, it represents how long an action is appropriate. 129

Supervised Autonomy An agent acts autonomously in the world while being monitored by a human. The human can select actions for the agent and the agent proposes actions it intends to execute to the human who has the power to preempt them. 50, 53, 58–60, 100, 128, 129

Acronyms

AI Artificial Intelligence. 1, 7, 134, 135

ANNs Artificial Neural Networks. 7

ASD Autism Spectrum Disorder. 13, 20, 58, 60, 96

CIDM 95% Confidence Interval of the Difference of the Mean. 70, 71

DREAM Development of Robot-Enhanced Therapy for Children with Autism Spectrum Disorders (European FP7 project). 9, 57

GUI Graphical User Interface. 61, 63, 65–67, 99, 109–111, 130, 137

HCI Human-Computer Interaction. 26, 29

HHI Human-Human Interaction. 14

HRC Human-Robot Collaboration. 17, 18, 31, 33, 54, 135

HRI Human-Robot Interaction. 1–3, 5, 9–11, 14, 19, 21, 23–27, 31–33, 36, 38–40, 45, 48, 58, 60, 92, 93, 96, 98, 101, 102, 105, 124, 127, 131, 132, 135, 137, 139, 140, 142

ILfD Interactive Learning from Demonstration. 43, 54

IML Interactive Machine Learning. 6, 9–11, 32, 34, 35, 39, 42, 45, 48, 51, 53, 54, 75, 76, 79, 95, 102, 130, 135, 142

IRL Interactive Reinforcement Learning. 6, 10, 42, 75–81, 83, 86–88, 90–97, 100–102, 140

LfD Learning from Demonstration. 28, 30, 32, 34–36, 39, 42, 44, 48, 54, 55, 100, 101, 125, 128

LfW Learning from the Wizard. 29

MDP Markov Decision Process. 37, 80

ML Machine Learning. 1, 3, 7, 17, 34, 40, 45, 49, 95, 97, 122–125, 133, 137, 138

NLP Natural Language Processing. 27

RAT Robot Assisted Therapy. 6, 12, 13, 22, 26, 30, 52, 58, 60, 72, 73, 102, 140

RL Reinforcement Learning. 7, 8, 34–41, 48, 52, 54, 55, 76, 96, 101, 135, 136, 142

SAR Socially Assistive Robotics. 12, 96, 101

SL Supervised Learning. 7, 35, 55, 76, 135

SPARC Supervised Progressively Autonomous Robot Competencies. 6–10, 48–55, 57–61, 65, 68–70, 72, 73, 75–81, 83, 86–103, 105–107, 112, 115, 121, 123–125, 127–142

WoZ Wizard-of-Oz. 6, 10, 26, 27, 29, 30, 52, 55, 57–61, 65, 68–70, 72, 100, 109, 121, 124, 128, 132

XAI Explainable Artificial Intelligence. 17

Appendices

Appendix A

Contribution to the DREAM Project

As part of the DREAM project, I have been involved in an international and multidisciplinary project. This led to numerous meetings and collaboration. While not presented in the main body of this thesis, I have contributed to a significant part of the software developed during the DREAM project, developed two tools to automate repetitive tasks and contributed to papers (Esteban et al., 2017; Cao et al., 2018).

A.1 Software Development

The main part of the development was focused on the architecture controlling a robot to provide therapeutic sessions for children with Autism Spectrum Disorder (ASD) (Esteban et al., 2017). Work Package 6 (Plymouth and Vrije Universiteit Brussel) was responsible to develop the robot's cognitive controller and implement the different tasks the robot would complete with the child. The overall system used the Supervised Autonomy: the robot suggests actions to the therapist, and the therapist can cancel them before an autonomous execution. In case of correct proposition, the therapist does not have to intervene, which leads to a reduction of the workload required for controlling the robot. If the proposed action is incorrect (due to sensor failure, unexpected child reaction or incorrect policy), the therapist can cancel it and manually select an alternative one. This ensures a therapeutically appropriate robot behaviour while not requiring the therapist to manually select each robot action.

I was the main developer for three components: the deliberative subsystem, the expression and actuation subsystem, and the naoInterface. The deliberative subsystem was responsible to interpret steps from a script defining the session flow, to keep track of the progress in the session and to select actions that should be submitted to the teacher. Then, based on the teacher approval or selection of an action, the actuation subsystem selected the primitives corresponding to the selected action and sent them to the robot interface. In the case of a Nao robot, the naoInterface converted these

general primitives into actions executable by the robot and used the robot API (here NaoQi) to make the robot act in the world.

The robot's behaviour was initially defined by human-readable scripts provided by the therapists. Part of the work was also to convert these scripts into series of steps the robot could follow. I had to interpret the therapists descriptions and translate them into logic steps including potential loops or branches (if statements). Then, I identified each simple bloc of behaviour and implemented the corresponding robot action in the deliberative, expression and actuation and naoInterface subsystems.

A.2 Tools

In addition to the components developed to run the DREAM application, I created two tools. The first one automated and simplified the use of the YARP middleware while conforming to the development standards imposed on the project, and the second one provided a human interface to create and modify the scripts used in the therapy.

A.2.1 yarpGenerator

The software development guidelines of DREAM imposed a specific folder structure for components, with a minimum number of 6 files with large overlap between components. Additionally, adding or removing one port for communication required changes in more than 10 locations in 5 files. To ease this procedure, I proposed to add two new classes: the yarpInterface and the componentController. The yarpInterface class exposes all the yarp output ports required by the component to C++ functions and integrates callbacks asynchronously called when messages arrive on input ports. The yarpController is a C++ only file where code can be developed without reliance on YARP and with easy access to the input and output ports. The yarpGenerator is a tool generating automatically compilable code, compliant with the development standards and including all the required files and the code to have any set of input and output ports working. Appendix B presents a technical report describing this tool more in depth.

A.2.2 scriptManager

The second tool provided a graphical way to read and edit the xml files describing the scripts used in the therapy scenario. This interface allows drag and drop of actions, updating parameters and copying scripts; however, no technical report is available.

Appendix B

yarpGenerator Technical Report

This appendix section presents the technical report corresponding to the yarpgenerator tool created for the DREAM project.



Development of Robot-enhanced Therapy for
Children with Autism Spectrum Disorders



Project No. 611391

DREAM
Development of Robot-enhanced Therapy for
Children with Autism Spectrum Disorders

TECHNICAL REPORT
Guideline For the YarpGenerator

Date: 18/05/2015

Technical report lead partner: **Plymouth University**

Primary Author: **E. Senft**

Revision: **1.4**

Project co-funded by the European Commission within the Seventh Framework Programme		
Dissemination Level		
PU	Public	
PP	Restricted to other programme participants (including the Commission Service)	PP
RE	Restricted to a group specified by the consortium (including the Commission Service)	
CO	Confidential, only for members of the consortium (including the Commission Service)	



Contents

Summary	2
Principal Contributors	2
Revision History	2
1 Yarp Component Generator	3
1.1 Concept	3
1.2 Utilisation	4
1.3 Integration in a Yarp project	5
1.4 Example component generation	6

Summary

This technical report presents the documentation for the second version of the YarpGenerator.

Principal Contributors

The main authors of this document are as follows (in alphabetical order).

Paul Baxter, Plymouth University
Emmanuel Senft, Plymouth University

Revision History

Version 1.0 (E.S. 03-02-2015)

Initial description

Version 1.1 (E.S. 12-05-2015)

Updated description to reflect changes in code

Version 1.2 (P.B. 13-05-2015)

Small clarifications etc.

Version 1.3 (E.S. 13-05-2015)

Small clarifications etc.

Version 1.4 (P.B. 18-05-2015)

Added example component generation.

Version 1.5 (E.S. 23-05-2016)

Update with new version of YarpGenerator and remove the Software Installation part.



1 Yarp Component Generator

1.1 Concept

In order to change a single port in a Yarp component, as specified in the software engineering standards, multiple parts of the code need to be modified, in different source files. But this process is a fixed one and can be automated. The purpose of the YarpGenerator is to provide an easy way to generate the structure of a component that conforms to the standards, including the different files and folders required, the code needed to use Yarp ports and the comments according to the standards.

This software enables the generation of compilable code, which can be added to a project (such as the DREAM integrated system) with minor additional modifications. Furthermore, it reduces the complexity of using Yarp ports by centralising the send/receive functions, and by providing if desired a simpler interface to access to the Yarp functionalities. For input ports, callbacks are used to enable event-based processing, without having to manually check for new information.

This software has been developed using Qt, is directly usable on windows and can be modified using QtCreator. This project contains five folders: components, descriptions, models, release and finally the sources.

The folder *components* is the folder where the generated components are located. A new folder with the component name is created and can be then just moved in the desired location.

Each time a component is generated, a description file with all the parameters needed for the generation is created in the folder *descriptions*. These files can be loaded in the software using the user interface by filing the lower text box with the name of the component followed by 'Description', as demonstrated when the software is started.

The folder *models* contains all the templates used to generate the other files. In the previous version, it appeared multiple times in the project, but this has been fixed, and now only one folder contains all the files used.

The folder *sources* contains all the sources used for the project and can be modified using Qt Creator. Some adjustment of the project have to be made to be able to run smoothly, such as changing the location of the build, the run command and so on.

Finally *release* contains the executable and the libraries needed by the tool.

As accepted in Brussels, the naming convention used for the port generation is: /componentname/portname:i-o.

This software will generate up to four folders and ten files. The files config.ini, name.h, name.cpp, nameConfiguration and nameMain.cpp are the ones defined in the Software Development Guide, and are required to allow the use of Yarp. In addition to these files described in the Software development guide, we propose to add four files to allow a simpler use of Yarp. The idea is to have the main code in nameController.cpp (with nameController.h as header) which does not require any Yarp functionalities, and a nameYarpInterface class providing a really simple interface between the main code and the files used for the Yarp communication. All the source files are located inside the src folder in the component.

In nameYarpInterface.cpp, each input port is linked to a function which will be called each time a message is received, and the message content is transmitted as a parameter. And then in this class, some simple conversions can be done to native c++ variables and then call the related function in nameController.cpp. The same idea is applied for output port: functions in Yarp interface can be called from the controller, do some conversion to Yarp type variables and send the message by calling functions in nameComputation.cpp. The other files (computation, configuration, main and the header) do not require to be modified for using Yarp.

1.2 Utilisation

Figure 1 shows the GUI of the YarpGenerator utility.

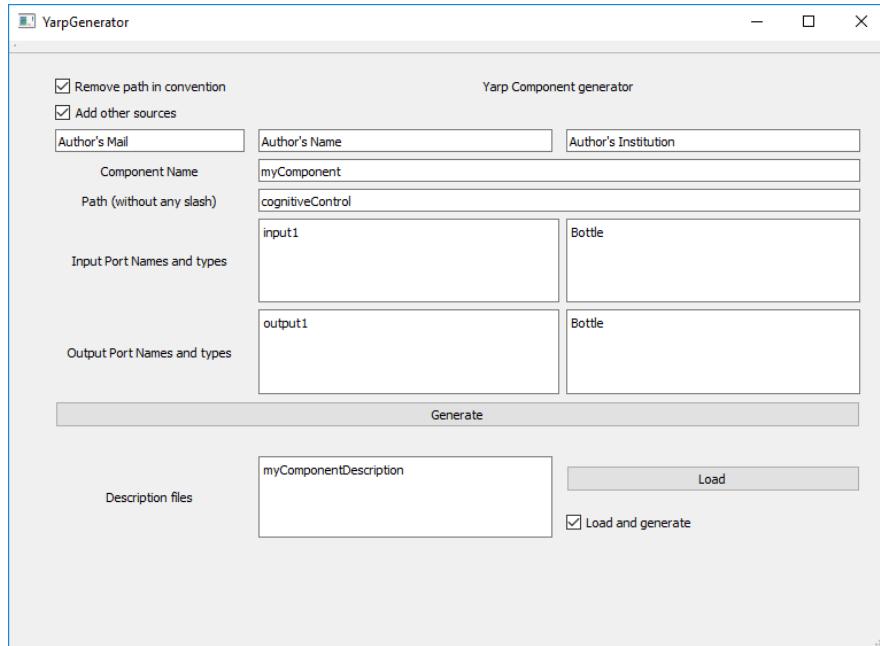


Figure 1: Interface of the YarpGenerator.

The first line of the interface contains information about the author, which will be directly included in the doxygen comments. The next two lines contain the name of the component and the path used for the port naming convention (not the generation). The first checkbox in the top right corner allows a switch to the DREAM convention. A second checkbox enables/disables the generation and use of the two additional classes proposed: `yarpInterface` and the `Controller`.

The next four boxes correspond to the different port names and their type (only yarp-compatible types should be used). In theory the number of port is not limited (i.e. a limit to the number of specified ports has not been encountered in testing), `BufferedPort` will be added in any case, so only `Bottle`, `VectorOf<int>`, `VectorOf<float>` and `VectorOf<double>` should be used.

The generate button generates the folder and files as described in figure 4. When a component is generated, a description file is created in the description folder and can be reused later to create a new component with the same ports and name. This can be useful if ports subsequently have to be renamed/added/removed after the first creation of the component: after regenerating the source files with the new port names, the yarp-standard interface code can be replaced without affecting the main computation source code. Only the functionalities required to get other information from the config files have to be added again.

The lower part of the interface can be used to generate components from the description file. Several files can be added in the text box and used at once. These description files should be put in the `/description` folder and follow this structure (instructions in italic and parameters in bold):

Remove path: **Value**

Add files: **Value**

Author's name: **Firstname Surname**



Author's institution: Institution's name

Author's mail: e-mail address

Component name: Name

Path: Path

Input Port Name and Type

input1 inputtype1

input2 inputtype2

Output Port Name and Type

output1 outputtype1

output2 outputtype2

The parameter **value** can take 0 or 1 and describes for the first line if the path should be removed when generating port names according to the convention selected (normally yes), and for the second line if the two secondary class should be created. A line should be inserted between the path line and the first input line, as well as between the last input line and the first output line. Similarly to the interface, only Yarp compliant types should be used. This is an example of description file content:

Remove path: 1

Add files: 1

Author's name: Emmanuel Senft

Author's institution: PLYM

Author's mail: emmanuel.senft@plymouth.ac.uk

Component name: myComponent

Path: cognitiveControl

Input Port Name and Type

in1 Bottle

in2 VectorOf<int>

in3 Bottle

Output Port Name and Type

out1 Bottle

out2 VectorOf<float>

1.3 Integration in a Yarp project

Once all the files are generated, only a few steps are required to be able to use the component in a Yarp project.

First, the main folder generated needs to be added in the project, and the CMakeList in /DREAM/release/components needs to be modified to include this new component and then rerun CMake to update the Visual Studio project 1.4. Then the main xml file describing the network (/DREAM/release/app/app.xml) needs to be changed: the new module needs to be added as well as each connection between its ports and the others specified.

1.4 Example component generation

The generation of a sample component is described in this section to demonstrate the main steps involved, and the main output files generated (with automatically generated contents). Figure 2 shows the definition of three ports for this example component (two input ports and one output port), and a text file name where the description is stored for later reload if required. The content of the description file is presented in figure 3.

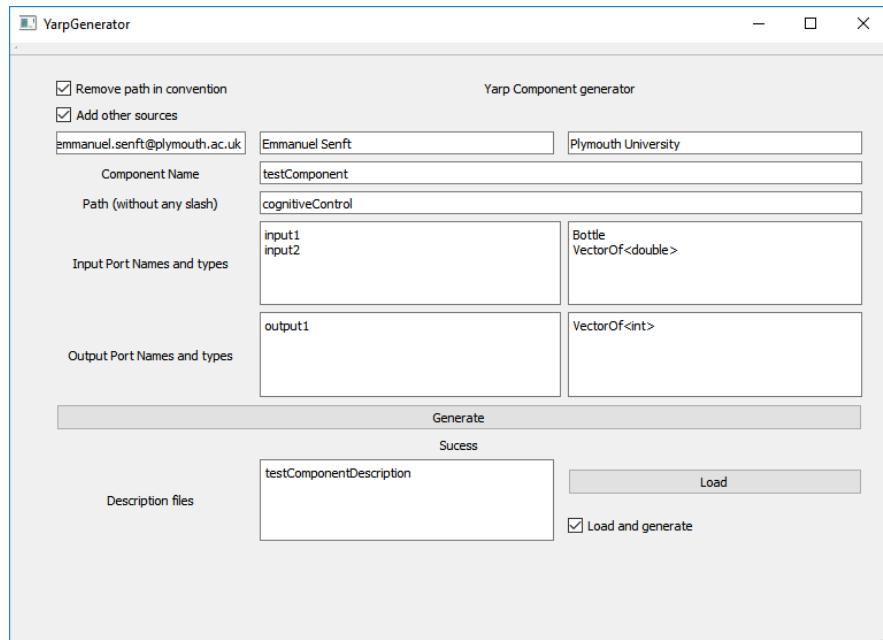


Figure 2: Definition of example component ports and types.

```
testComponentDescription.txt
1 Remove path: 1
2 Add files: 1
3 Author's name: Emmanuel Senft
4 Author's institution: Plymouth University
5 Author's mail': emmanuel.senft@plymouth.ac.uk
6 Component name: testComponent
7 Path: cognitiveControl
8
9 Input Port Name and Type
10 input1 Bottle
11 input2 VectorOf<double>
12
13 Output Port Name and Type
14 output1 VectorOf<int>
```

Figure 3: Description file associated with the example presented in figure 2.

Generating a new component with these settings results in a file structure as shown in figure 4. Unlike the previous version, now a flat structure is adopted for the files, all the sources (the 4 mandatory and the 4 optional) are in the same folder: src. Note also that the xml test application is not automatically generated at the moment, and must be made manually.

In addition to the source code contents, the yarpGenerator utility also automatically generates the

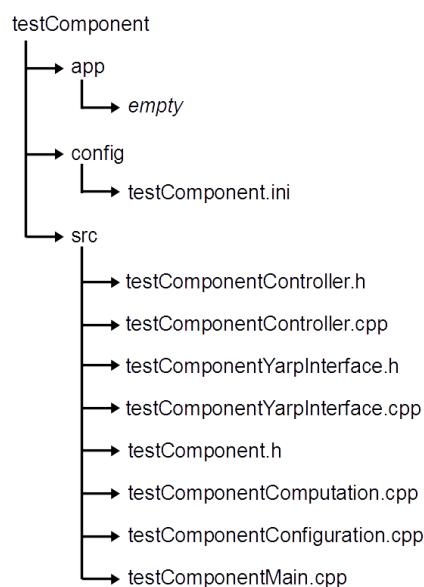


Figure 4: File structure created for the testComponent. An xml application file is not created automatically; in the current version of the yarpGenerator, this has to be created manually. This is the same structure as described in the wiki.

Doxxygen-compatible comments in the header of the testComponent.h file based on the instantiated ports. A sample of this is shown in figure 5.

```

* <b>Configuration File Parameters </b>
*
* The following key-value pairs can be specified as parameters in the configuration file
* (they can also be specified as command-line parameters if you so wish).
* The value part can be changed to suit your needs; the default values are shown below.
*
* Key | Value
* :-- | :---
* _input1In | /testComponent/input1:i
* _input2In | /testComponent/input2:i
*
* - description
*
* Key | Value
* :-- | :---
* _output1Out | /testComponent/output1:o
*
* - description
*
* \section ports\_sec Ports Accessed
*
* - None
*
* \section ports\_sec Ports Created
*
* <b>Input ports</b>
*
* - \c /testComponent
*
* - \c /testComponent/input1:i
* - \c /testComponent/input2:i
*
* <b>Output ports</b>
*
* - \c /testComponent
*
* - \c /testComponent/output1:o
*
* <b>Port types </b>
*
* The functional specification only names the ports to be used to communicate with the component
* but doesn't say anything about the data transmitted on the ports. This is defined by the following code.
*
* \c BufferedPort<Bottle>      input1In;
* \c BufferedPort<VectorOf<double>>      input2In;
* \c BufferedPort<VectorOf<int>>      output1Out;

```

Figure 5: Sample of the automatically generated comments in the main test component header, showing the port definitions.

Appendix C

Social Assistive Robot for Cardiac Rehabilitation

A second project I was involved in during this work was a collaboration with the Escuela Colombiana de Ingeniería Julio Garavito in Bogota. This project: the Human-Robot Interaction Strategies for Rehabilitation based on Socially Assistive Robotics project (Royal Academy of Engineering: IAPP\1516\137) aimed at developing a social robot to support, monitor and assist cardiac rehabilitation for patients affected or susceptible to suffer from cardiovascular disease.

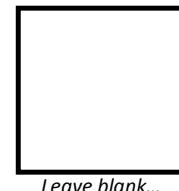
During that project, my role was to design the robot's controller. The robot guided the patients through the therapy, provided encouragements or feedback on their postures, monitored their vital signs (blood pressure, heart rate and pain level) and alerted the therapists if the patients were in a bad situation. We used a combination of preprocessed inputs and image processing with rules from the therapists to define a robot behaviour suited to this therapy. So far this collaboration contributed to two publications (Lara et al., 2017; Casas et al., 2018), but more are being written.

Appendix D

Information and Questionnaires for the First Study

This appendix section presents the information sheet and questionnaires used in the first study. Participants first completed the demographic questionnaire, then they were given the information sheet describing the task before watching a short video of the interface and interacting with the two systems. After each interaction they completed the post-interaction questionnaire and the experiment finished with the post-experiment questionnaire comparing the two conditions. The instructional video demonstrating the interface and the files can be found at <https://emmanuel-senft.github.io/experiment-woz.html>.

DEMOGRAPHICS



Leave blank...

1. Sex

 Female Male

2. Age: _____

3. Are you right or left handed?

 Left Right

4. How familiar with social robots are you?

(please circle one)

Not at all	Slightly	Somewhat	Moderately	Extremely
------------	----------	----------	------------	-----------

5. Have you acted as a 'wizard' in an HRI experiment before?

 Yes No

6. How familiar are you with tutoring children/students on a one-to-one basis?

(in contrast to lecturing which is typically to groups)

(please circle one)

Not at all	Slightly	Somewhat	Moderately	Extremely
------------	----------	----------	------------	-----------

EXPERIMENT INFORMATION

Thank you for agreeing to take part in this pilot study. Please take your time to read this information, and please do refer back to it if desired while the experiment is in progress.

The aim of the experiment is to assess the effectiveness of an algorithm that learns how to teach while being responsive to the motivation and engagement state of the learner. Your role in this is to supervise the *teacher robot* to enable it to learn an effective teaching strategy.

The setup you will be asked to supervise consists of two robots: a *child robot* and a *teacher robot*. The robot facing you is the *child robot*, which needs to learn about the difference between sad and happy faces by playing a sorting game on the touchscreen. It displays a range of behaviours that indicate its levels of motivation and engagement, including gaze behaviour, speed of categorisation movements, etc.

In order for the *child robot* to learn, it is necessary for its engagement and motivation levels to be sufficiently high (although they will naturally vary over time). The *teacher robot* must therefore provide suitable encouragement and feedback to enable this. If the teacher behaves inappropriately, this will have an adverse effect on the engagement and motivation of the *child robot*. The behaviour of the *child robot* is determined by its engagement and motivation levels.

The robot with its back to you is the *teacher robot*. This is the robot you are supervising. Every so often, it suggests an action to take. If this action is inappropriate, then your task is to provide an alternative suitable action to take. The *teacher robot* will learn from this in order to improve its future suggestions.

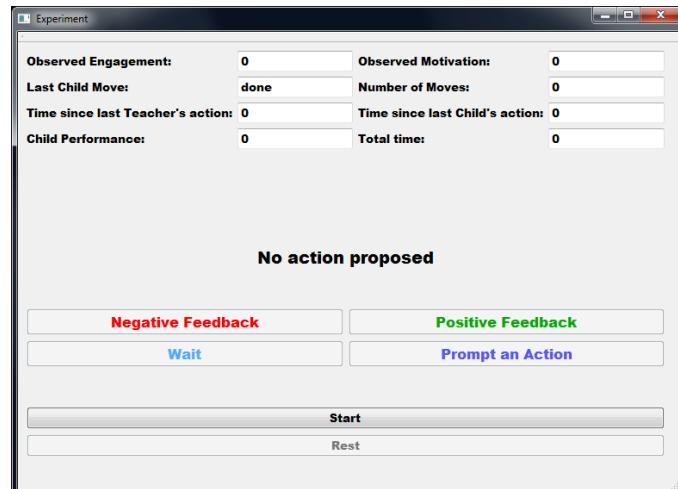
You will have a range of information available to you to help determine what the ideal *teacher robot* behaviour should be. It is up to you to decide what information you use and how you use it.

On the following page you will find further information on how to supervise the *teacher robot* through the GUI, and some behavioural characteristics of the *child robot*.

You will be asked to supervise two different *teacher robots* in two separate interactions, with the same *child robot* (learning and states reset at the beginning of each interaction). Each interaction will last for 10 minutes, after which the *teacher robot* will end the interaction.

The Supervisor GUI

The GUI provides some overall information on the child's behaviour (last move, time since last move), and an estimate of the *child robot* engagement and motivation states. This is only an estimate based on a subset of characteristics that are mainly updated after each *child robot* move – however, this estimate is used by the *teacher robot*



learning algorithm. As a supervisor, you are also able to see the *child robot* behaviour, which is not directly accessible by the *teacher robot* (the engagement determines gaze behaviour and frequency of moves, and the motivation determines speed of movement and the success of moves). The following actions are available:

Wait	Do nothing at the moment
Prompt an Action	Encourage the <i>child robot</i> to do an action
Positive Feedback	Congratulate the <i>child robot</i> on making a correct classification
Negative Feedback	Feedback (supportive) for an incorrect classification

Every so often, the *teacher robot* will suggest an action to you (in the centre of the GUI). You will have 3 seconds to decide whether it is appropriate. If it is, then no action is required from you: the *teacher robot* will execute the suggested action. If you think there is a better action, then choose one of the four options within the 3 second period.

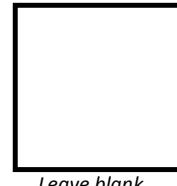
The *child robot* model

The engagement and motivation levels of the child robot are modified by its own actions, and those of the teacher robot. Generally, prompting an action causes the *child robot* Engagement to increase. The effect of *child robot* classification move and *teacher robot* feedback on the *child robot* Engagement (E) and Motivation (M) is as follows:

	+ive feedback	-ive feedback
Good move	E↑ M↑	E↓ M↓
Bad move	E↑ M↓	E↓ M↑
No move	E↓ M↓	E↓ M↓

One final effect is that the Engagement and Motivation values have a maximum limit: if this is approached, then there is a chance that further actions to increase them will actually result in a sharp decrease.

POST-INTERACTION



Leave blank...

1. The “child robot” learned during the interaction...

(please circle one)

Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
-------------------	----------	-------------------	----------------------------	----------------	-------	----------------

2. The performance of the “child robot” improved in response to the “teacher robot” actions...

(please circle one)

Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
-------------------	----------	-------------------	----------------------------	----------------	-------	----------------

3. The “teacher robot” is capable of making appropriate action decisions in future interactions without supervision...

(please circle one)

Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
-------------------	----------	-------------------	----------------------------	----------------	-------	----------------

4. The “teacher robot” always suggested an incorrect or inappropriate action...

(please circle one)

Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
-------------------	----------	-------------------	----------------------------	----------------	-------	----------------

5. By the end of the interaction, my work-load was very light...

(please circle one)

Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
-------------------	----------	-------------------	----------------------------	----------------	-------	----------------

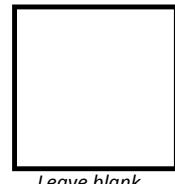
6. What did you pay most attention to during the interaction?

(please circle one)

Child Robot	Touchscreen	GUI	Other (please state)
-------------	-------------	-----	----------------------

Other: _____

POST-EXPERIMENT



Leave blank...

- 1. There was a clear difference in behaviour between the two “teacher robots”...*

(please circle one)

Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
-------------------	----------	-------------------	----------------------------	----------------	-------	----------------

- 2. There was a clear difference in behaviour between the two “child robots”...*

(please circle one)

Strongly disagree	Disagree	Somewhat disagree	Neither agree nor disagree	Somewhat agree	Agree	Strongly agree
-------------------	----------	-------------------	----------------------------	----------------	-------	----------------

- 3. Which “teacher robot” was better able to perform the task?*

(please circle one)

 First Second

- 4. Which “teacher robot” did you prefer supervising?*

(please circle one)

 First Second

- 5. What information did you use to make your decisions in the interactions?*

(please circle all that apply)

Child Robot behaviour	Touchscreen feedback	GUI data	Other (please state)
-----------------------	----------------------	----------	----------------------

Other: _____

Please turn over...

6. Do you have any comments regarding the teacher/child robot setup?

7. Do you have any other general comments regarding the experiment and/or methodology?

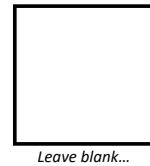
Thank you for your help!

Appendix E

Information and Questionnaires for the Second Study

This appendix section presents the information sheets and questionnaires used in the second study. Participants first completed the demographic questionnaire and were given the information sheet describing the overall task. Then, they were provided with the description sheet of the system they would interact first with (depending of their group). After this first interaction with a system (consisting of 3 sessions), participants completed a first time the post-interaction questionnaire, then were presented with the other information sheet, interacted with the other condition and completed a second post-interaction questionnaire. Similarly to the first study, the experiment finished with the post-experiment questionnaire comparing the two conditions. These files can be found at <https://emmanuel-senft.github.io/experiment-irl.html>.

DEMOGRAPHICS



Leave blank...

1. *Gender:*

Female

Male

2. *Age:* ____

3. *Occupation:*

4. *Background/Course:*

5. *How familiar with machine learning are you?*

(please circle one)

Not at all	Slightly	Somewhat	Moderately	Extremely
------------	----------	----------	------------	-----------

6. *How familiar with social robots are you?*

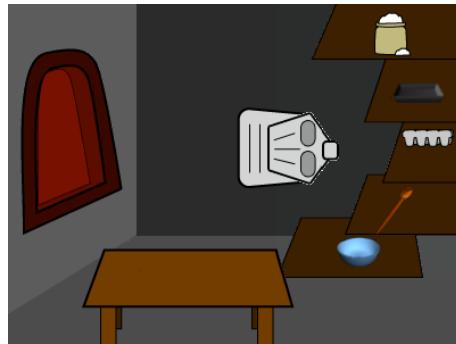
(please circle one)

Not at all	Slightly	Somewhat	Moderately	Extremely
------------	----------	----------	------------	-----------

Introduction

Welcome to Sophie's Kitchen. This is an environment developed by Andrea Thomaz. In this experiment, you will interact with Sophie, a robot, and teach her how to bake a cake in a kitchen. You will test two different ways of communicating with her: (1) using feedback and guidance, or (2) through a system for suggestion/correction of actions.

The Kitchen Environment



In this task, Sophie is in the kitchen environment pictured. There are five objects in the kitchen: flour, eggs, a bowl, a spoon, and a tray. Sophie can move to one of three locations: to face the shelf on the right, the oven on the left, or the table in the centre. The five objects can be in any of these locations as well.

Sophie's Actions

- She can TURN-LEFT or TURN-RIGHT, which changes her location.
- She can PICK-UP any object in her current location, she can only hold one object at a time.
- She can PUT-DOWN the object she is holding.
- She can USE the object she is holding on any object in her current location, for example:
 - Using the flour on the bowl, puts flour in the bowl, or
 - Using the spoon on the bowl with flour and eggs makes batter.

The Kitchen Task: Bake a cake

Sophie does not know how to bake a cake, your task is to help her learn how to! The basic steps are: put the bowl on the table, add flour, eggs, and stir to make batter, put the batter in the tray, and put the tray in the oven. Sophie may need help making the cake a few times before she can do it herself.

Disasters and Goals

Sometimes Sophie will accidentally do actions that lead to the Disaster state. (Like putting the spoon in the oven!) When this happens “Disaster” will flash on the screen, the kitchen gets cleaned up and Sophie starts a new practice round. Additionally, if Sophie successfully bakes the cake, “Goal!” will flash on the screen, the kitchen gets cleaned up and Sophie starts a new practice round. For the disaster state, Sophie is automatically sent a negative message. For the goal state, Sophie is automatically sent a positive message.

System A

With this system, you can't do actions or tell Sophie exactly what to do, but you can send messages to try and help. When you click the LEFT mouse button a rectangle appears, showing your message. Drag the mouse to change the size and colour of your message. UP = GREEN (positive), DOWN = RED (negative).

Guidance Messages

In addition to the feedback, you can direct Sophie's attention to objects with guidance messages. Click the RIGHT mouse button to make a yellow square. This square tells Sophie 'Pay attention to this!'. Objects light up when the mouse is over them to help you know what guidance message you will send. You can only guide Sophie to an object (not a location like the table), and she only sees your message if she is facing the object. For example, if she is facing the table and you make the yellow square over the flour on the shelf she won't see that, but if she's facing the table and the flour is on the table she will see your guidance message and think you are telling her to pay attention to or do something with the flour.

Task

You will have to teach Sophie how to bake the cake and test her skill. This task should be repeated three times. She will have forgotten what happened each time, although she will learn in the same way each time.

End of the interaction

When you think that Sophie knows how to bake a cake well enough, or when you don't want to interact with her anymore, you can press the "Sophie is Ready!" button, which will make Sophie try to bake a cake on her own to test her knowledge. You won't be able to correct anything in that case. If she does an action ending in a disaster state or if she takes more than 70 actions to bake the cake, the learning task is counted as failed. If you clicked on the button because you don't want to interact anymore, please indicate this in the questionnaire at the end of the interaction.

System B

Sophie will propose the action she wants to execute by gazing in the direction of an object or a location, and then give you some time to correct her if you would prefer to see her execute a different action.

Communication

She will propose an action by looking at an object or a place. By looking toward an object, she means that she is about to do an action with it. If she looks to an empty space, she plans to put down the object she is carrying. If she looks at a different location, she plans to move there. Based on her plan, you have multiple ways to respond. You can wait and let Sophie execute the action she initially planned. But you can also guide her actions if you want. Click the RIGHT mouse button to make a yellow square. This square tells Sophie 'Pay attention to this!'. Objects or locations light up when the mouse is over them to help you know what guidance message you will send. You can guide Sophie to an object or a location, and she only sees your message on object if she is facing the object or if the message relates to a location. For example, if she is facing the table and you make the yellow square over the flour on the shelf she won't see that, but if she's facing the table and the flour is on the table she will see your guidance message and think you are telling her to pick up the flour.

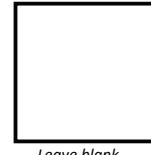
Task

You will have to teach Sophie how to bake the cake and test her skill. This task should be repeated three times. She will have forgotten what happened each time, although she will learn in the same way each time.

End of the interaction

When you think that Sophie knows how to bake a cake well enough, or when you don't want to interact with her anymore, you can press the "Sophie is Ready!" button, which will make Sophie try to bake a cake on her own to test her knowledge. You won't be able to correct anything in that case. If she does an action ending in a disaster state or if she takes more than 70 actions to bake the cake, the learning task is counted as failed. If you clicked on the button because you don't want to interact anymore, please indicate this in the questionnaire at the end of the interaction.

POST INTERACTION



Leave blank...

Please place an "X" along each scale at the point that best indicates your experience with the system.

Mental Demand: How much mental and perceptual activity was required (e.g., thinking, deciding, calculating, remembering, looking, searching, etc.)? Was the mission easy or demanding, simple or complex, exacting or forgiving?



Physical Demand: How much physical activity was required (e.g., pushing, pulling, turning, controlling, activating, etc.)? Was the mission easy or demanding, slow or brisk, slack or strenuous, restful or laborious?



Temporal Demand: How much time pressure did you feel due to the rate or pace at which the mission occurred? Was the pace slow and leisurely or rapid and frantic?



Performance: How successful do you think you were in accomplishing the goals of the mission? How satisfied were you with your performance in accomplishing these goals?



Effort: How hard did you have to work (mentally and physically) to accomplish your level of performance?



Frustration: How discouraged, stressed, irritated, and annoyed versus gratified, relaxed, content, and complacent did you feel during your mission?



How many times did you press the button “Sophie is Ready” because you did not want to interact anymore?

(please circle one)

0	1	2	3
---	---	---	---

Please rate these following statements:

1. I felt I had control over the robot’s actions.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

2. My actions had an important impact on the robot’s behaviour.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

3. It was difficult to control the robot.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

4. I did not have enough control of the robot.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

5. It was easy to teach the robot.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

6. The interaction was natural.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

7. The robot had clear intentions concerning the actions it wanted to execute.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

8. The interaction with the robot was transparent.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

9. The robot was predictable.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

10. I had a clear oversight of the interaction.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

11. I knew when the robot was about to do an incorrect action.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

12. The way to supervise the robot was efficient.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

13. The way to supervise the robot was of high quality.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

14. Now I trust the robot to continue the interaction on its own.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

15. I trusted the robot to do the right thing at the right time.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

16. The robot was trustworthy.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

Please rate your impression of the robot on these scales:

Incompetent	<input type="checkbox"/>	Competent				
Ignorant	<input type="checkbox"/>	Knowledgeable				
Irresponsible	<input type="checkbox"/>	Responsible				
Unintelligent	<input type="checkbox"/>	Intelligent				
Foolish	<input type="checkbox"/>	Sensible				
Dislike	<input type="checkbox"/>	Like				
Unfriendly	<input type="checkbox"/>	Friendly				
Unkind	<input type="checkbox"/>	Kind				
Unpleasant	<input type="checkbox"/>	Pleasant				
Awful	<input type="checkbox"/>	Nice				
Fake	<input type="checkbox"/>	Natural				
Machinelike	<input type="checkbox"/>	Humanlike				
Unconscious	<input type="checkbox"/>	Conscious				
Artificial	<input type="checkbox"/>	Lifelike				
Moving rigidly	<input type="checkbox"/>	Moving elegantly				

POST EXPERIMENT



Leave blank...

1. Which robot did you prefer interacting with?

(please circle one)

First

Second

Why?

2. Which robot learned better?

First

Second

Why?

3. Which robot performed better?

First

Second

Why?

4. Which of the interactions was the more natural?

First

Second

Why?

5. How often did you pay attention to the first robot's gaze?

Never	Rarely	Occasionally	Almost every time	Every Time
-------	--------	--------------	-------------------	------------

6. The first robot's gazing behaviour was useful.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

7. How often did you pay attention to the second robot's gaze?

Never	Rarely	Occasionally	Almost every time	Every Time
-------	--------	--------------	-------------------	------------

8. The second robot's gazing behaviour was useful.

Strongly Disagree	disagree	neutral	Agree	Strongly Agree
-------------------	----------	---------	-------	----------------

9. Do you have any comments regarding the first robot?

10. Do you have any comments regarding the second robot?

11. Do you have any other general comments regarding the experiment?

12. In the future, how would you like to teach a robot?

Appendix F

Teacher's Diary

This appendix section presents the daily report of the teacher's impressions and feelings when teaching the robot in the study in Chapter 6. It should be noted that among all the children supervised, many were special needs and as such have been removed from the result analysis. This also explains the difference of number between the children in the supervised condition ($n=25$) and in this diary ($n=34$).

Research Diary

N.B.: Record of my role in Emmanuel's Robot's for Learning Study

02 February 2018

My Role as Teacher

In order to test the usability of Emmanuel's system I will play the role of naive teacher to train the robot's behaviours. I am unfamiliar with how the system operates and have never used it before or anything similar. This Diary will denote my experiences as I learn to use the system to inform conclusions on how easy the system might be for real-world teachers.

Training with the System - no participants

Lots going on. It's difficult to pay attention to everything. Hardest aspect is probably moving animals around. We ran autonomous to see what the robot would do with training so far, it was mainly Encouragement and Congratulations. Very chatty. Need to make sure that my training encourages more silence and more suggestions for moving animals.

05 February 2018

Training and Passive Condition

Ran participants in the passive condition only so that I could continue to watch the activity on the tablet and get used to looking at everything that's happening.

Tablet froze in round 3 of game. Did not restart by its self. Not a WiFi issue as robot was still working.

Main focus when watching the passive condition is where I would move the animals to promote learning. This may be made slightly more difficult by having to track and change the robot's verbal feedback.

Problems with Tablet connecting to wifi - several restarts. Emmanuel says it's fairly normal.

Training on SPARC with Emmanuel as student (x2): Goals - try lots of actions and try to mediate the robot's timing using the skip and cancel buttons so it learns to leave silence gaps. Robot learns very quickly. Made a couple of mistakes - need to make sure I always select object and target before suggesting how to move an animal. Robot gets quite excited about its learned actions - will start making lots of suggestions rapidly. Need to pay attention to it a lot. The hardest part is keeping up with the participant/student.

06 February 2018

Training with non-participant and participant

Non-Participant with SPARC - Blank Robot. I skipped/cancelled a lot of the robot's suggestions, including ones which I wanted to approve. The robot's suggestion-execution speed takes getting used to. Spent a lot of time moving animals around. Tried to remember to use verbal feedback too. Tablet crashed during the final game session. Robot only learns exactly what I've shown it so I need to make sure I show it all the possible connections between animals. Teaching method: used demonstration during first game sessions, then started using "draw attention" more in the later sessions to see if the participant knew what to do with the animals. Also used "congratulations" and "encouragement" both when the participant followed robot's instructions correctly, and for a lot of their own correct actions which they performed without the robot's help.

Participant with SPARC - (Blank) Learning Robot.

Participant 1 - Slow to start, seemed like low confidence or not sure what to do. Followed instruction/demonstration well. Easy to work with. Lots of animal movement demonstrations as participant did not initiate many movements by themselves.

Participant 2 - aggressive play. I found it difficult to know how best to respond. Tried to reduce number of demonstrations and increase the use of "drawing attention" in later game sessions. Had to use "remind rules" a lot and found it difficult to suggest animal movements.

Participant 3 - gradually used a more aggressive play style. Responded fairly well to demonstration. Tried to reduce number of demonstrations and increase the use of "drawing attention" in later game sessions. Used "remind rules" a lot when play became more aggressive.

Teaching experience is fun but tiring. I find I'm dismissing robot suggestions more than I actually want to - some are valid but I am "playing safe" by skipping/cancelling all in order to avoid inappropriate suggestions. Need to trust the robot more. This does require more effort though as I need to pay attention to the robot suggestions better.

07 February 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 4 - Tablet crashed part way through game 2. Let participant make first move on each game. Tried to have a more balanced mix of demonstrations, drawing attention and verbal behaviours. Allowed the robot to perform more suggested actions, e.g. demonstrations. Robot suggests to start talking very early in each session (before participant's first move) so will try to teach it to wait. Remainder of game sessions were done in passive mode due to tablet crash.

Participant 5 - Achieving a better balance between my own actions and robot's suggestions. Sometimes the number of suggestions from the robot is a bit overwhelming. I try to use participants' performance on pre and mid tests to inform my teaching. Robot performed one action 3 times in succession which was very distracting and it threw me off a bit. Participant employed a waiting tactic in the last game session

so had to ensure that the robot didn't perform any actions until the participant was ready.

08 February 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 6 - Started with an aggressive style of play so in sessions 1 and 2 I used "remind rules" a bit. Also started out using some demonstrations because they had made a few wrong connections in pre-test. Play style was also hectic, they would move animals around quickly and see what happened when they interacted with other animals. This was difficult to work with - often the animal I wanted to do a demonstration with or draw attention to was being used by the time the robot behaviour started. In sessions 3 and 4 I used a few demonstrations but mostly "encouragement" and "congratulations". I cancelled most of the robot suggestions due to how hectic the participant's behaviour was. Participant also employed a waiting tactic at the beginning of later sessions so robot behaviours were always cancelled until the participants' first move.

09 February 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 7 - aggressive play = keeps eating all "raw foods" at once. Makes game play difficult. Used "remind rules" a lot. Found it difficult to give demonstrations but managed some. Allowed some robot suggestions but not many as I wanted to slow game-play down.

Participant 8 - allowing more robot suggestions. Feel like I have a good balance between demonstrations and vocalisations. Didn't need to use "remind rules". Participant seemed to want to work more independently, occasionally finishing their own task before attending to the robot's suggestions. I used fewer demonstrations in the later sessions.

Participant 9 - slow deliberative play style. Participant is whispering to themselves about the animals and what they might eat - possible memory technique. I am not offering too many robot behaviours but am using a good balance of demonstrations and verbal feedback. Not using "remind rules". Participant employed waiting tactic in last round. Robot also waited before giving suggestions. Not sure if it waited for first move or if it just waited for a certain amount of time.

Participant 10 - stopped after first game session so was excluded. I used quite a few demonstrations. Participant was good at working things out for themselves though. When completing the game after the participant left the robot was autonomously giving encouragement followed by "remind rules" when Emmanuel was just eating animals to end the game. Robot was only giving 2 responses when Emmanuel was quick-finishing the game and I feel this is good spacing between responses - not too chatty. I will try to continue teaching the robot to space out the instructions/interactions.

19 February 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 11 - haphazard play style, sometimes hard to keep up. Ended up accidentally making a mistake in my demonstration. Need to make sure I don't hold the animals too long before moving them. Participant is very talkative so am trying to get the robot to behave as if it has heard him. Slowed down my responses in the second half. Was able to provide more useful behaviours. Had to use quite a few demonstrations and remind rules.

Participant 12 - the tablet crashed just before the end of game session 1. Participant was excluded. Up until then I had done a number of demonstrations. Cancelled most but not all robot behaviours (mainly "remind rules" as participant seemed comfortable with what was needed). Participant also got a lot of interactions right first time so used congratulations and encouragement a bit.

22 February 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 13 - Careful and slow play style. Tried not to make too many suggestions as participant as they were using an exploratory style to find out what animals ate. In sessions 3+4 participant adopted an aggressive style, using up all an animal's energy in one go. Difficult to work with, participant did not respond to "remind rules" and it was hard to give demonstrations.

Participant 14 - not responding to demonstrations in session 1, i.e. wouldn't feed the animal when I moved it to its food. Seemed to get better at this in session 2. I gave quite a few demos and only congratulated if they were followed through. Participant would often take food to the animal (so animal noises were "ouch"). Still congratulated this as they did it following demonstrations to feed target animals.

Participant 15 - Used a lot of demonstrations in session 1 but fewer in session 2. Participant adopted exploratory style but was also quite aggressive = using up all raw food resources quickly. One demo they weren't sure of so tried again in session 3 and 4, participant worked it out. Participant got really good at the game so mostly I just congratulated and encouraged. Let the robot carry out a lot of its suggested behaviours and only did demos so that the robot wasn't just talking but was also playing with the participant

Participant 16 - Very quick learner. Did a couple of demonstrations but not many. Will try to use more robot suggestions as robot was often suggested good things but I was auto-skipping them. Focused on the more difficult connections, e.g. dragonfly eats other bugs, frogs eat more than just flies. They seem to learn this in the game.

Participant 17 - Very talkative so easy to know what help to give. Learns quickly. Was frustrated with the fact that there's a point where you can't keep going because you run out of raw food. Was talking with the robot a lot and was excited about learning new things. Very fun and easy to teach with demonstrations.

16 April 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 18 - talks about what they're doing so easy to offer direction. Says things like "I don't know what xx eats". Also does some exploration so often it was not necessary to give demonstrations. Picked up the game very quickly. Adopted the waiting tactic. Last game was mostly encouragement and congratulations.

Participant 19 - Seems to have a good grip on the game but plays at a relatively slow rate. Doesn't speak whilst playing. Mainly encouragement and congratulations with occasional demonstration. Also using draw attention more as they seem to know what they're doing but struggle to attend to all the animals. Adopted a destructive approach in the 3rd session. Started using remind rules more. Allowed quite a few robot suggestions. One went a bit weird but otherwise they were all good. It's now fairly easy to control the robot. The hardest thing is when participant is struggling to work out what an animal eats but you can't give a demonstration because they won't let the animal go.

Participant 20 - participant struggled so gave lots of demonstrations. Took them a while to understand the purpose of the demonstrations. Would eat all raw foods in one go. Chose not to use remind rules yet though as they were struggling to work out what each animal ate. Started using remind rules in sessions 3 and 4. They started picking up the rules in session 3 but reverted to destructive tactics in session 4.

Note on teaching experience: It's often frustrating that I can't pick exactly what the robot says, but at the same time, if I could there would definitely be too many buttons on the screen. The current layout is already difficult to get to grips with so there's not much that can be done about this.

17 April 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 21 - (excluded) very quiet but seemed to learn quickly. Made moves slowly and was nervous of making guesses or playing in an exploratory way so I did a lot of demonstrations and congratulations to build confidence. Controlling the robot is really easy now, although I still tend not to let it carry out its suggested actions even when they are valid.

Participant 22 - Allowed the robot to carry out more of its suggestions. Participant is very quiet and seems unsure of themselves/what to do so used a lot of demonstrations. Tried not to overwhelm them though as they seem to prefer moving at a slower pace. Seemed to get better at the game, but I'm not sure how much that has to do with my demonstrations. There is a risk of the teacher/robot taking over the task rather than just supporting the child's learning. This is especially true in cases where the participant does not like to explore/try out pairings.

Participant 23 - (excluded) was very unsure of what the animals would eat to start so encouraged exploration. They took on a destructive style of play so started using remind rules during session 3. Participant gave up.

18 April 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 24 - Was fairly exploratory to start. Used a fair few demonstrations and let the robot carry out a lot of suggestions. Robot was trying to remind rules following correct actions though. Not sure why but I will try to stop it from doing this. Participant seemed to get the hang of the game towards the end but would start every trial in an exploratory way - not remembering what animals ate what food? During test seemed to just connect animals fairly randomly but with some thought about what might make sense. Appeared to be making connections for the sake of making connections whilst still trying to be right, i.e. would start drawing an arrow then hold it in place until they decided on a food.

Participant 25 - Wasn't very confident to start (only 1 connection in first test). Gained confidence after a couple of demonstrations and some encouragement from the robot. Made a lot more connections in second test and was muttering (possible indicator of using memory of the game). Was much more confident in final test.

Participant 26 - Happy to explore. I gave (what I thought was) an even balance of demo, encourage and congratulate. Let the robot carry out a lot of its suggestions. Have started using "cancel" button more, mainly for "remind rules". In session 3 and 4 I didn't need to do much, they have a good handle on the game and a good technique. Mainly encourage and congratulate, and more "draw attention" than I would otherwise.

19 April 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 27 - (excluded) I find it takes me the first session to work out what kind of behaviours the robot should perform so session 1 is often a bit slow. Participant seems confident with game rules but unsure of connections between animals and food so using demonstrations. Participant also uses aggressive play style so gradually incorporating more remind rules behaviours.

Participant 28 - Participant has a semi-aggressive style of play. Doesn't tend to eat all of a food source in one go but gets close occasionally. Sometimes when the robot congratulates correct moves participants seem to repeat the action immediately in response to this reward. The encouragement phrases sometimes have the same effect, especially ones like keep going. Play style got more aggressive in later sessions so used remind rules and intervening demonstrations more.

Participant 29 - Participant barely needed me. Almost every demonstration I tried they were already performing. Mostly just encourage and congratulate needed. However, seems to act a bit too slow for the game pace when left to own devices. Added a few more demonstrations in session 4 but it was difficult to synchronize with both the participant's pace and the game.

20 April 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 30 - (excluded) Seemed quite scared when the robot moved so kept demonstrations to a minimum. Participant got used to the robot in session 2 and even looked at the robot waiting for it to give a suggestion. Not sure if looking at the robot for help or because bored but used this as a cue to give directions/re-engage participant with game. This seemed effective.

Participant 31 - (excluded) talked through everything they did. Didn't need much help but did demos just to make the experience fun. Understood the game very well and played in a systematic way. Made a wrong action when trying to perform action on animal participant was holding. Not much can be done in these situations as sometimes we just grab the same animal at the same time.

Participant 32 - Quite chatty so fairly easy to give appropriate direction. I kept missing the skip button so a few erroneous behaviours slipped through. Participant's play style got very hectic towards the end so it was difficult to keep up. Ended up doing mostly vocal-only behaviours.

23 April 2018

Training with participant

Participant with SPARC - Learning Robot.

Participant 33 - Seemed nervous to start but picked up the game after a few demonstrations. Adopted a fairly aggressive style but did not do remind rules just yet. Robot did not suggest remind rules either. Not sure if this is good or bad. Used remind rules in sessions 3 and 4, participant seemed to apply these suggestions.

Participant 34 - Aggressive and haphazard play style to start, difficult to get them to slow down. Very chatty. Used remind rules quite a few times in the first 2 sessions. Found it easier to let the participant do most of the actions, whilst the robot just encouraged, mainly because the participants style was so hectic it was hard to keep up. Participant probably needed input to tell them to slow down both during the game and the test, they seemed to confuse themselves by trying to do everything quickly.

Participant with Autonomous.

Participant 1 - Participant seemed very capable and I felt that the robot was too active during session 1. Not sure why as I usually try not to do much during session 1 unless the participant seems unsure or nervous. 2nd session went better, the robot was less chaotic. 3rd session felt a bit more hectic with the robot performing behaviours which weren't necessary (e.g. draw attention to dragonfly immediately after demonstrating bird-dragonfly). Robot is using remind rules seemingly randomly - I wanted it to use it when the participant is aggressively eating when the animal does not need to.

Bibliography

- Abbeel, P., & Ng, A. Y. (2004). Apprenticeship Learning Via Inverse Reinforcement Learning. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*.
- Adams, J. A., Rani, P., & Sarkar, N. (2004). Mixed Initiative Interaction and Robotic Systems. In *AAAI Workshop on Supervisory Control of Learning and Adaptive Systems*, (p. 613).
- Alili, S., Alami, R., & Montreuil, V. (2009). A Task Planner for an Autonomous Social Robot. In *Distributed Autonomous Robotic Systems 8*, (pp. 335–344). Springer.
- Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., & Topcu, U. (2017). Safe Reinforcement Learning Via Shielding. *ArXiv Preprint ArXiv:1708.08611*.
- Amershi, S., Cakmak, M., Knox, W. B., & Kulesza, T. (2014). Power to the People: the Role of Humans in Interactive Machine Learning. *AI Magazine*, 35(4), 105–120.
- Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A Survey of Robot Learning From Demonstration. *Robotics and Autonomous Systems*, 57(5), 469–483.
- Arkin, R. C., Fujita, M., Takagi, T., & Hasegawa, R. (2003). An Ethological and Emotional Basis for Human-Robot Interaction. *Robotics and Autonomous Systems*, 42(3-4), 191–201.
- Asada, H., Slotine, J.-J., & Slotine, J.-J. (1986). *Robot Analysis and Control*. John Wiley & Sons.
- Bartneck, C., & Forlizzi, J. (2004). A Design-Centred Framework for Social Human-Robot Interaction. In *Proceedings of the 13th IEEE International Workshop on Robot and Human Interactive Communication*, (pp. 31–33).
- Barto, A. G., & Mahadevan, S. (2003). Recent Advances in Hierarchical Reinforcement Learning. *Discrete Event Dynamic Systems*, 13(1-2), 41–77.
- Basu, C., Singhal, M., & Dragan, A. D. (2018). Learning From Richer Human Guidance: Augmenting Comparison-Based Learning With Feature Queries. In *Proceedings of the 13th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 132–140). ACM.
- Baxter, P., Ashurst, E., Kennedy, J., Senft, E., Lemaignan, S., & Belpaeme, T. (2015a). The Wider Supportive Role of Social Robots in the Classroom for Teachers. In *Proceedings of the 1st International Workshop on Educational Robotics, at ICSR*.
- Baxter, P., Kennedy, J., Senft, E., Lemaignan, S., & Belpaeme, T. (2016). From Characterising Three Years of HRI to Methodology and Reporting Recommendations. In *Proceedings of the 11th ACM/IEEE International Conference on Human Robot Interation (HRI)*, (pp. 391–398). IEEE Press.
- Baxter, P., Matu, S., Senft, E., Costescu, C., Kennedy, J., David, D., & Belpaeme, T. (2015b). Touchscreen-Mediated Child-Robot Interactions Applied to ASD Therapy. In *Proceedings of the 1st International Conference on Social Robots in Therapy and Education*. Almere, Netherlands.

- Baxter, P., Wood, R., & Belpaeme, T. (2012). A Touchscreen-Based ‘Sandtray’ to Facilitate, Mediate and Contextualise Human-Robot Social Interaction. In *Proceedings of the 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 105–106).
- Beer, J., Fisk, A. D., & Rogers, W. A. (2014). Toward a Framework for Levels of Robot Autonomy in Human-Robot Interaction. *Journal of Human-Robot Interaction*, 3(2), 74.
- Beetz, M., Kirsch, A., & Muller, A. (2004). RPLLEARN: Extending an Autonomous Robot Control Language to Perform. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, (pp. 1022–1029). IEEE Computer Society.
- Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton.
- Belpaeme, T., Baxter, P. E., Read, R., Wood, R., Cuayáhuitl, H., Kiefer, B., Racioppa, S., Kruijff-Korbayová, I., Athanasopoulos, G., Enescu, V., et al. (2012). Multimodal Child-Robot Interaction: Building Social Bonds. *Journal of Human-Robot Interaction*, 1(2), 33–53.
- Belpaeme, T., Kennedy, J., Ramachandran, A., Scassellati, B., & Tanaka, F. (2018). Social Robots for Education: a Review. *Science Robotics*, 3(21), eaat5954.
- Bennewitz, M., Faber, F., Joho, D., Schreiber, M., & Behnke, S. (2005). Towards a Humanoid Museum Guide Robot That Interacts With Multiple Persons. In *5th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, (pp. 418–423). IEEE.
- Billard, A., Calinon, S., Dillmann, R., & Schaal, S. (2008). Robot Programming by Demonstration. In *Springer Handbook of Robotics*, (pp. 1371–1394). Springer.
- Bloom, B. S. (1984). The 2 Sigma Problem: the Search for Methods of Group Instruction as Effective as One-to-One Tutoring. *Educational Researcher*, 13(6), 4–16.
- Bouton, M. E. (2007). *Learning and Behavior: a Contemporary Synthesis*. Sinauer Associates.
- Bowling, M., Burch, N., Johanson, M., & Tammelin, O. (2015). Heads-Up Limit Hold'em Poker is Solved. *Science*, 347(6218), 145–149.
- Bray, T. (2009). *Confronting the Shadow Education System: What Government Policies for What Private Tutoring?*. United Nations Educational, Scientific and Cultural Organization; International Institute for Educational Planning.
- Breazeal, C. (1998). A Motivational System for Regulating Human-Robot Interaction. In *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, (pp. 54–62). AAAI.
- Breazeal, C. L. (2004). *Designing Sociable Robots*. MIT press.
- Brooks, R. A. (1986). A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation*, 2(1), 14–23.
- Brunskill, E., & Li, L. (2014). Pac-Inspired Option Discovery in Lifelong Reinforcement Learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, (pp. 316–324).

- Burgard, W., Cremers, A. B., Fox, D., Hähnel, D., Lakemeyer, G., Schulz, D., Steiner, W., & Thrun, S. (1999). Experiences With an Interactive Museum Tour-Guide Robot. *Artificial Intelligence*, 114(1-2), 3–55.
- Cakmak, M., Chao, C., & Thomaz, A. L. (2010). Designing Interactions for Robot Active Learners. *IEEE Transactions on Autonomous Mental Development*, 2(2), 108–118.
- Cao, H.-L., Esteban, P. G., Simut, R., Van de Perre, G., Lefeber, D., Vanderborght, B., et al. (2017). A Collaborative Homeostatic-Based Behavior Controller for Social Robots in Human-Robot Interaction Experiments. *International Journal of Social Robotics*, 9(5), 675–690.
- Cao, H.-L., Van de Perre, G., Kennedy, J., Senft, E., Esteban, P. G., De Beir, A., Simut, R., Belpaeme, T., Lefeber, D., & Vanderborght, B. (2018). A Personalized and Platform-Independent Behavior Control System for Social Robots in Therapy: Development and Applications. *IEEE Transactions on Cognitive and Developmental Systems*.
- Casas, J., Irfan, B., Senft, E., Gutiérrez, L., Rincon-Roncancio, M., Munera, M., Belpaeme, T., & Cifuentes, C. A. (2018). Social Assistive Robot for Cardiac Rehabilitation: a Pilot Study With Patients With Angioplasty. In *Companion of the 13th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 79–80). ACM.
- Casper, J., & Murphy, R. R. (2003). Human-Robot Interactions During the Robot-Assisted Urban Search and Rescue Response at the World Trade Center. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 33(3), 367–385.
- Chao, C., Cakmak, M., & Thomaz, A. L. (2010). Transparent Active Learning for Robots. In *Proceedings of the 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 317–324). IEEE.
- Chernova, S., & Veloso, M. (2009). Interactive Policy Learning Through Confidence-Based Autonomy. *Journal of Artificial Intelligence Research*, 34(1).
- Clark-Turner, M., & Begum, M. (2018). Deep Reinforcement Learning of Abstract Reasoning From Demonstrations. In *Companion of the 13th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 372–372). ACM.
- Cohen, P. A., Kulik, J. A., & Kulik, C.-L. C. (1982). Educational Outcomes of Tutoring: a Meta-Analysis of Findings. *American Educational Research Journal*, 19(2), 237–248.
- Cooper, J. O., Heron, T. E., Heward, W. L., et al. (2007). *Applied Behavior Analysis*. Pearson/Merrill-Prentice Hall Upper Saddle River, NJ.
- Cover, T., & Hart, P. (1967). Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, 13(1), 21–27.
- Dautenhahn, K. (1999). Robots as Social Actors: AuRoRa and the Case of Autism. In *Proceedings of CT99, the 3rd International Cognitive Technology Conference*, vol. 359, (p. 374).
- Dautenhahn, K. (2004). Robots We Like to Live With?!-A Developmental Perspective on a Personalized, Life-Long Robot Companion. In *13th IEEE International Workshop on Robot and Human Interactive Communication (RO-MAN)*, (pp. 17–22). IEEE.
- Dautenhahn, K., & Werry, I. (2004). Towards Interactive Robots in Autism Therapy: Background, Motivation and Challenges. *Pragmatics & Cognition*, 12(1), 1–35.

- Di Nuovo, A., Broz, F., Belpaeme, T., Cangelosi, A., Cavallo, F., Esposito, R., & Dario, P. (2014). A Web Based Multi-Modal Interface for Elderly Users of the Robot-Era Multi-Robot Services. In *IEEE International Conference on Systems, Man and Cybernetics (SMC)*, (pp. 2186–2191). IEEE.
- Diehl, J. J., Schmitt, L. M., Villano, M., & Crowell, C. R. (2012). The Clinical Use of Robots for Individuals With Autism Spectrum Disorders: a Critical Review. *Research in Autism Spectrum Disorders*, 6(1), 249–262.
- Dienes, Z. (2011). Bayesian Versus Orthodox Statistics: Which Side Are You On? *Perspectives on Psychological Science*, 6(3), 274–290.
- Dragan, A. D., Lee, K. C., & Srinivasa, S. S. (2013). Legibility and Predictability of Robot Motion. In *Proceedings of the 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 301–308). IEEE.
- Esteban, P. G., Baxter, P., Belpaeme, T., Billing, E., Cai, H., Cao, H.-L., Coeckelbergh, M., Costescu, C., David, D., De Beir, A., et al. (2017). How to Build a Supervised Autonomous System for Robot-Enhanced Therapy for Children With Autism Spectrum Disorder. *Paladyn, Journal of Behavioral Robotics*, 8(1), 18–38.
- Fails, J. A., & Olsen Jr, D. R. (2003). Interactive Machine Learning. In *Proceedings of the 8th International Conference on Intelligent User Interfaces*, (pp. 39–45). ACM.
- Faulkner, T. K., Short, E. S., & Thomaz, A. L. (2018). Policy Shaping With Supervisory Attention Driven Exploration. In *Proceedings of the IEEE/RSJ 2018 International Conference on Intelligent Robots and Systems (IROS)*.
- Feil-Seifer, D., & Matarić, M. J. (2005). Defining Socially Assistive Robotics. In *9th International Conference on Rehabilitation Robotics (ICORR)*, (pp. 465–468). IEEE.
- Fincannon, T., Barnes, L. E., Murphy, R. R., & Riddle, D. L. (2004). Evidence of the Need for Social Intelligence in Rescue Robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 2, (pp. 1089–1095). IEEE.
- Fink, J., Bauwens, V., Kaplan, F., & Dillenbourg, P. (2013). Living With a Vacuum Cleaning Robot. *International Journal of Social Robotics*, 5(3), 389–408.
- Fong, T., Nourbakhsh, I., & Dautenhahn, K. (2003). A Survey of Socially Interactive Robots. *Robotics and Autonomous Systems*, 42(3-4), 143–166.
- Frager, S., & Stern, C. (1970). Learning by Teaching. *The Reading Teacher*, 23(5), 403–417.
- Friedman, B., Kahn Jr, P. H., & Hagman, J. (2003). Hardware Companions?: What Online AIBO Discussion Forums Reveal About the Human-Robotic Relationship. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, (pp. 273–280). ACM.
- García, J., & Fernández, F. (2015). A Comprehensive Survey on Safe Reinforcement Learning. *Journal of Machine Learning Research*, 16, 1437–1480.
- Gehring, W. J., Goss, B., Coles, M. G., Meyer, D. E., & Donchin, E. (1993). A Neural System for Error Detection and Compensation. *Psychological Science*, 4(6), 385–390.

- Gockley, R., Bruce, A., Forlizzi, J., Michalowski, M., Mundell, A., Rosenthal, S., Sellner, B., Simmons, R., Snipes, K., Schultz, A. C., et al. (2005). Designing Robots for Long-Term Social Interaction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (pp. 1338–1343). IEEE.
- Goodrich, M. A., & Schultz, A. C. (2008). Human-Robot Interaction: a Survey. *Foundations and Trends in Human-Computer Interaction*, 1(3), 203–275.
- Gordon, G., Spaulding, S., Westlund, J. K., Lee, J. J., Plummer, L., Martinez, M., Das, M., & Breazeal, C. (2016). Affective Personalization of a Social Robot Tutor for Children's Second Language Skills. In *AAAI*, (pp. 3951–3957).
- Griffith, S., Subramanian, K., Scholz, J., Isbell, C., & Thomaz, A. L. (2013). Policy Shaping: Integrating Human Feedback With Reinforcement Learning. In *Advances in Neural Information Processing Systems*, (pp. 2625–2633).
- Grollman, D. H., & Jenkins, O. C. (2007). Dogged Learning for Robots. In *International Conference on Robotics and Automation (ICRA)*, (pp. 2483–2488). IEEE.
- Guizzo, E., & Ackerman, E. (2012). How Rethink Robotics Built Its New Baxter Robot Worker. *IEEE Spectrum*, (p. 18).
- Han, J., Jo, M., Park, S., & Kim, S. (2005). The Educational Use of Home Robots for Children. In *IEEE International Workshop on Robot and Human Interactive Communication (RO-MAN)*, (pp. 378–383). IEEE.
- Hart, S. G. (2006). NASA-Task Load Index (NASA-TLX); 20 Years Later. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 50, (pp. 904–908). Sage Publications Sage CA: Los Angeles, CA.
- Hart, S. G., & Staveland, L. E. (1988). Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. *Advances in Psychology*, 52, 139–183.
- Harwin, W., Ginige, A., & Jackson, R. (1988). A Robot Workstation for Use in Education of the Physically Handicapped. *IEEE Transactions on Biomedical Engineering*, 35(2), 127–131.
- Hawes, N., Burbridge, C., Jovan, F., Kunze, L., Lacerda, B., Mudrová, L., Young, J., Wyatt, J., Hebesberger, D., Kortner, T., et al. (2017). The STRANDS Project: Long-Term Autonomy in Everyday Environments. *IEEE Robotics & Automation Magazine*, 24(3), 146–156.
- Hayes, B., & Shah, J. A. (2017). Improving Robot Controller Transparency Through Autonomous Policy Explanation. In *Proceedings of the 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 303–312). ACM.
- Hood, D., Lemaignan, S., & Dillenbourg, P. (2015). When Children Teach a Robot to Write: an Autonomous Teachable Humanoid Which Uses Simulated Handwriting. In *Proceedings of the 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 83–90). ACM.
- Howard, R. A. (1960). *Dynamic Programming and Markov Processes*. MIT Press.
- Irfan, B., Kennedy, J., Lemaignan, S., Papadopoulos, F., Senft, E., & Belpaeme, T. (2018). Social Psychology and Human-Robot Interaction: an Uneasy Marriage. In *Companion of the 13th ACM/IEEE International Conference on Human-Robot Interaction (Alt.HRI)*.

- Isbell, C. L., Kearns, M., Singh, S., Shelton, C. R., Stone, P., & Kormann, D. (2006). Cobot in LambdaMOO: an Adaptive Social Statistics Agent. *Autonomous Agents and Multi-Agent Systems*, 13(3), 327–354.
- Jain, A., Wojcik, B., Joachims, T., & Saxena, A. (2013). Learning Trajectory Preferences for Manipulators Via Iterative Improvement. In *Advances in Neural Information Processing Systems*, (pp. 575–583).
- JASP Team (2018). JASP (Version 0.8.6)[Computer Software].
URL <https://jasp-stats.org/>
- Jeffreys, H. (1998). *The Theory of Probability*. OUP Oxford.
- Johnson, D. W., Johnson, R. T., & Smith, K. A. (1991). *Active Learning: Cooperation in the College Classroom*. Interaction Book Company Edina, MN.
- Joorabchi, M. E., Mesbah, A., & Kruchten, P. (2013). Real Challenges in Mobile App Development. In *2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, (pp. 15–24). IEEE.
- Kanda, T., Glas, D. F., Shiomi, M., Ishiguro, H., & Hagita, N. (2008). Who Will Be the Customer?: a Social Robot That Anticipates People's Behavior From Their Trajectories. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, (pp. 380–389). ACM.
- Kanda, T., Hirano, T., Eaton, D., & Ishiguro, H. (2004). Interactive Robots as Social Partners and Peer Tutors for Children: a Field Trial. *Human-Computer Interaction*, 19(1), 61–84.
- Kanda, T., Shiomi, M., Miyashita, Z., Ishiguro, H., & Hagita, N. (2009). An Affective Guide Robot in a Shopping Mall. In *Proceedings of the 4th ACM/IEEE International Conference on Human Robot Interaction (HRI)*, (pp. 173–180). ACM.
- Kaochar, T., Peralta, R. T., Morrison, C. T., Fasel, I. R., Walsh, T. J., & Cohen, P. R. (2011). Towards Understanding How Humans Teach Robots. In *Proceedings of the International Conference on User Modeling, Adaptation, and Personalization*, (pp. 347–352). Springer.
- Kasparov, G. (2010). The Chess Master and the Computer. *The New York Review of Books*, 57(2), 16–19.
- Kelley, J. F. (1983). An Empirical Methodology for Writing User-Friendly Natural Language Computer Applications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, (pp. 193–196). ACM.
- Kennedy, J., Baxter, P., & Belpaeme, T. (2015a). The Robot Who Tried Too Hard: Social Behaviour of a Robot Tutor Can Negatively Affect Child Learning. In *Proceedings of the 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 67–74). ACM.
- Kennedy, J., Baxter, P., Senft, E., & Belpaeme, T. (2015b). Higher Nonverbal Immediacy Leads to Greater Learning Gains in Child-Robot Tutoring Interactions. In *Proceedings of the International Conference on Social Robotics (ICSR)*, (pp. 327–336). Springer International Publishing.
- Kennedy, J., Baxter, P., Senft, E., & Belpaeme, T. (2015c). Using Immediacy to Characterise Robot Social Behaviour in Child-Robot Interactions. In *Proceedings of the 1st Workshop on Evaluating Child-Robot Interaction, at ICSR*.

- Kennedy, J., Baxter, P., Senft, E., & Belpaeme, T. (2016a). Heart vs Hard Drive: Children Learn More From a Human Tutor Than a Social Robot. In *Proceedings of the 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI) Extended Abstracts*, (pp. 451–452). IEEE.
- Kennedy, J., Baxter, P., Senft, E., & Belpaeme, T. (2016b). Social Robot Tutoring for Child Second Language Learning. In *Proceedings of the 11th ACM/IEEE International Conference on Human Robot Interation (HRI)*, (pp. 231–238). IEEE Press.
- Kennedy, J., Lemaignan, S., Montassier, C., Lavalade, P., Irfan, B., Papadopoulos, F., Senft, E., & Belpaeme, T. (2017). Child Speech Recognition in Human-Robot Interaction: Evaluations and Recommendations. In *Proceedings of the 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 82–90). ACM.
- Knox, W. B., Spaulding, S., & Breazeal, C. (2014). Learning Social Interaction From the Wizard: a Proposal. In *Workshops at the 28th AAAI Conference on Artificial Intelligence*.
- Knox, W. B., Spaulding, S., & Breazeal, C. (2016). Learning From the Wizard: Programming Social Interaction Through Teleoperated Demonstrations. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems (AAMAS)*, (pp. 1309–1310). International Foundation for Autonomous Agents and Multiagent Systems.
- Knox, W. B., & Stone, P. (2009). Interactively Shaping Agents Via Human Reinforcement: the TAMER Framework. In *Proceedings of the 5th International Conference on Knowledge Capture*, (pp. 9–16). ACM.
- Knox, W. B., & Stone, P. (2010). Combining Manual Feedback With Subsequent MDP Reward Signals for Reinforcement Learning. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, (pp. 5–12).
- Kober, J., Bagnell, J. A., & Peters, J. (2013). Reinforcement Learning in Robotics: a Survey. *The International Journal of Robotics Research*, 32(11), 1238–1274.
- Kucukyilmaz, A., & Demiris, Y. (2018). Learning Shared Control by Demonstration for Personalized Wheelchair Assistance. *IEEE Transactions on Haptics*.
- Lara, J. S., Casas, J., Aguirre, A., Munera, M., Rincon-Roncancio, M., Irfan, B., Senft, E., Belpaeme, T., & Cifuentes, C. A. (2017). Human-Robot Sensor Interface for Cardiac Rehabilitation. In *Proceedings of the International Conference on Rehabilitation Robotics (ICORR)*, (pp. 1013–1018). IEEE.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep Learning. *Nature*.
- Leite, I., Martinho, C., & Paiva, A. (2013). Social Robots for Long-Term Interaction: a Survey. *International Journal of Social Robotics*, 5(2), 291–308.
- Lemaignan, S., Edmunds, C. E. R., Senft, E., & Belpaeme, T. (2018). The PlInSoRo Dataset: Supporting the Data-Driven Study of Child-Child and Child-Robot Social Dynamics. *PLOS ONE*, 13(10), 1–19.
URL [Https://doi.org/10.1371/journal.pone.0205999](https://doi.org/10.1371/journal.pone.0205999)
- Lemaignan, S., Garcia, F., Jacq, A., & Dillenbourg, P. (2016). From Real-Time Attention Assessment to With-Me-Ness in Human-Robot Interaction. In *Proceedings of the 11th ACM/IEEE International Conference on Human Robot Interaction (HRI)*, (pp. 157–164). IEEE Press.

- Leyzberg, D., Spaulding, S., & Scassellati, B. (2014). Personalizing Robot Tutors to Individuals' Learning Differences. In *Proceedings of the 9th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 423–430). ACM.
- Leyzberg, D., Spaulding, S., Toneva, M., & Scassellati, B. (2012). The Physical Presence of a Robot Tutor Increases Cognitive Learning Gains. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 34.
- Linder, S. P., Nestruck, B. E., Mulders, S., & Lavelle, C. L. (2001). Facilitating Active Learning With Inexpensive Mobile Robots. In *Journal of Computing Sciences in Colleges*, vol. 16, (pp. 21–33). Consortium for Computing Sciences in Colleges.
- Liu, P., Glas, D. F., Kanda, T., Ishiguro, H., & Hagita, N. (2014). How to Train Your Robot-Teaching Service Robots to Reproduce Human Social Behavior. In *23rd IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, (pp. 961–968).
- Loftin, R., Peng, B., MacGlashan, J., Littman, M. L., Taylor, M. E., Huang, J., & Roberts, D. L. (2016). Learning Behaviors Via Human-Delivered Discrete Feedback: Modeling Implicit Feedback Strategies to Speed Up Learning. *Autonomous Agents and Multi-Agent Systems*, 30(1), 30–59.
- MacGlashan, J., Ho, M. K., Loftin, R., Peng, B., Wang, G., Roberts, D. L., Taylor, M. E., & Littman, M. L. (2017). Interactive Learning From Policy-Dependent Human Feedback. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*.
- Mason, M., & Lopes, M. C. (2011). Robot Self-Initiative and Personalization by Learning Through Repeated Interactions. In *Proceedings of the 6th International Conference on Human-Robot Interaction (HRI)*, (pp. 433–440). ACM.
- Matarić, M. J., Eriksson, J., Feil-Seifer, D. J., & Weinstein, C. J. (2007). Socially Assistive Robotics for Post-Stroke Rehabilitation. *Journal of NeuroEngineering and Rehabilitation*, 4(1), 5.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-Level Control Through Deep Reinforcement Learning. *Nature*, 518(7540), 529–533.
- Montreuil, V., Clodic, A., Ransan, M., & Alami, R. (2007). Planning Human Centered Robot Activities. In *IEEE International Conference on Systems, Man and Cybernetics (ISIC)*, (pp. 2618–2623). IEEE.
- Moray, N. (2013). *Mental Workload: Its Theory and Measurement*, vol. 8. Springer Science & Business Media.
- Mubin, O., Stevens, C. J., Shahid, S., Al Mahmud, A., & Dong, J.-J. (2013). A Review of the Applicability of Robots in Education. *Journal of Technology in Education and Learning*, 1(209-0015), 13.
- Munzer, T., Toussaint, M., & Lopes, M. (2017). Efficient Behavior Learning in Human-Robot Collaboration. *Autonomous Robots*, (pp. 1–13).
- Murphy, R. R. (2004). Human-Robot Interaction in Rescue Robotics. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 34(2), 138–153.
- Murphy, R. R., Tadokoro, S., Nardi, D., Jacoff, A., Fiorini, P., Choset, H., & Erkmen, A. M. (2008). Search and Rescue Robotics. In *Springer Handbook of Robotics*, (pp. 1151–1173). Springer.

- Nevidjon, B., & Erickson, J. I. (2001). The Nursing Shortage: Solutions for the Short and Long Term. *Online Journal of Issues in Nursing*, 6(1), 4.
- Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., & Ng, A. Y. (2009). ROS: an Open-Source Robot Operating System. In *ICRA Workshop on Open Source Software*, vol. 3, (p. 5). Kobe, Japan.
- Rasmussen, J., & Vicente, K. J. (1989). Coping With Human Errors Through System Design: Implications for Ecological Interface Design. *International Journal of Man-Machine Studies*, 31(5), 517–534.
- Riek, L. (2012). Wizard of Oz Studies in HRI: a Systematic Review and New Reporting Guidelines. *Journal of Human-Robot Interaction*, 1(1), 119–136.
- Rieser, V., & Lemon, O. (2008). Learning Effective Multimodal Dialogue Strategies From Wizard-of-Oz Data: Bootstrapping and Evaluation. *Proceedings of ACL-08: HLT*, (pp. 638–646).
- Russell, S. J., & Norvig, P. (2016). *Artificial Intelligence: a Modern Approach*. Malaysia; Pearson Education Limited.,
- Salter, T., Dautenhahn, K., & Bockhorst, R. (2004). Robots Moving Out of the Laboratory—Detecting Interaction Levels and Human Contact in Noisy School Environments. In *13th IEEE International Workshop on Robot and Human Interactive Communication (RO-MAN)*, (pp. 563–568). IEEE.
- Scheutz, M., Krause, E., Oosterveld, B., Frasca, T., & Platt, R. (2017). Spoken Instruction-Based One-Shot Object and Action Learning in a Cognitive Robotic Architecture. In *Proceedings of the 16th Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, (pp. 1378–1386). International Foundation for Autonomous Agents and Multiagent Systems.
- Scheutz, M., Schermerhorn, P., Kramer, J., & Anderson, D. (2007). First Steps Toward Natural Human-Like HRI. *Autonomous Robots*, 22(4), 411–423.
- Senft, E., Baxter, P., & Belpaeme, T. (2015a). Human-Guided Learning of Social Action Selection for Robot-Assisted Therapy. In *Proceedings of the Workshop on Machine Learning for Interactive Systems (MLIS'15)*, at ICML, (pp. 15–20). JMLR Workshop & Conference Series.
- Senft, E., Baxter, P., Kennedy, J., & Belpaeme, T. (2015b). SPARC: Supervised Progressively Autonomous Robot Competencies. In *Proceedings of the International Conference on Social Robotics (ICSR)*, (pp. 603–612). Springer.
- Senft, E., Baxter, P., Kennedy, J., & Belpaeme, T. (2015c). When is It Better to Give Up?: Towards Autonomous Action Selection for Robot Assisted ASD Therapy. In *Proceedings of the 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI) Extended Abstracts*, (pp. 197–198). ACM.
- Senft, E., Baxter, P., Kennedy, J., Lemaignan, S., & Belpaeme, T. (2016a). Providing a Robot With Learning Abilities Improves Its Perception by Users. In *Proceedings of the 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI) Extended Abstracts*, (pp. 513–514). IEEE Press.
- Senft, E., Baxter, P., Kennedy, J., Lemaignan, S., & Belpaeme, T. (2017a). Supervised Autonomy for Online Learning in Human-Robot Interaction. *Pattern Recognition Letters*, 99, 77–86.

- Senft, E., Lemaignan, S., Bartlett, M., Baxter, P., & Belpaeme, T. (2018a). Robots in the Classroom: Learning to Be a Good Tutor. In *Proceedings of the 4th Workshop on Robots for Learning (R4L) - Inclusive Learning, at HRI*.
- Senft, E., Lemaignan, S., Baxter, P., & Belpaeme, T. (2017b). Toward Supervised Reinforcement Learning With Partial States for Social HRI. In *Proceedings of the Artificial Intelligence for Human-Robot Interaction Symposium (AI-HRI), at AAAI Fall Symposium Series*.
- Senft, E., Lemaignan, S., Baxter, P., & Belpaeme, T. (2018b). From Evaluating to Teaching: Rewards and Challenges of Human Control for Learning Robots . In *Proceedings of the Workshop on Human/Robot in the Loop Machine Learning (HRML), at IROS*.
- Senft, E., Lemaignan, S., Baxter, P. E., & Belpaeme, T. (2016b). SPARC: an Efficient Way to Combine Reinforcement Learning and Supervised Autonomy. In *Proceedings of the Future of Interactive Learning Machines Workshop (FILM), at NIPS*.
- Senft, E., Lemaignan, S., Baxter, P. E., & Belpaeme, T. (2017c). Leveraging Human Inputs in Interactive Machine Learning for Human Robot Interaction. In *Proceedings of the Companion of the 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, (pp. 281–282). ACM.
- Sequeira, P., Alves-Oliveira, P., Ribeiro, T., Di Tullio, E., Petisca, S., Melo, F. S., Castellano, G., & Paiva, A. (2016). Discovering Social Interaction Strategies for Robots From Restricted-Perception Wizard-of-Oz Studies. In *Proceedings of the 11th ACM/IEEE International Conference on Human Robot Interation (HRI)*, (pp. 197–204). IEEE Press.
- Settles, B. (2009). Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin-Madison.
- Sheridan, T. B. (2016). Human–Robot Interaction: Status and Challenges. *Human Factors*, 58(4), 525–532.
- Sheridan, T. B., & Verplank, W. L. (1978). Human and Computer Control of Undersea Teleoperators. Tech. rep., MIT CAMBRIDGE MAN-MACHINE SYSTEMS LAB.
- Sherif, M. (1936). *The Psychology of Social Norms*. Harper.
- Shiomi, M., Sakamoto, D., Kanda, T., Ishi, C. T., Ishiguro, H., & Hagita, N. (2008). A Semi-Autonomous Communication Robot: a Field Trial at a Train Station. In *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction (HRI)*, (pp. 303–310). ACM.
- Singer, P. W. (2009). *Wired for War: the Robotics Revolution and Conflict in the 21st Century*. Penguin.
- Soh, H., & Demiris, Y. (2015). Spatio-Temporal Learning With the Online Finite and Infinite Echo-State Gaussian Processes. *IEEE Transactions on Neural Networks and Learning Systems*, 26(3), 522–536.
- Sutton, R. S., & Barto, A. G. (1998). *Reinforcement Learning: an Introduction*. MIT press.
- Tanaka, F., & Matsuzoe, S. (2012). Children Teach a Care-Receiving Robot to Promote Their Learning: Field Experiments in a Classroom for Vocabulary Learning. *Journal of Human-Robot Interaction*, 1(1).

- Tapus, A., Mataric, M. J., & Scassellati, B. (2007). Socially Assistive Robotics [grand Challenges of Robotics]. *IEEE Robotics & Automation Magazine*, 14(1), 35–42.
- Theocharous, G., Thomas, P. S., & Ghavamzadeh, M. (2015). Personalized Ad Recommendation Systems for Life-Time Value Optimization With Guarantees. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI)*, (pp. 1806–1812).
- Thill, S., Pop, C. A., Belpaeme, T., Ziemke, T., & Vanderborght, B. (2012). Robot-Assisted Therapy for Autism Spectrum Disorders With (Partially) Autonomous Control: Challenges and Outlook. *Paladyn*, 3(4), 209–217.
- Thomaz, A. L., & Breazeal, C. (2008). Teachable Robots: Understanding Human Teaching Behavior to Build More Effective Robot Learners. *Artificial Intelligence*, 172(6), 716–737.
- Thrun, S., Bennewitz, M., Burgard, W., Cremers, A. B., Dellaert, F., Fox, D., Hahnel, D., Rosenberg, C., Roy, N., Schulte, J., et al. (1999). MINERVA: a Second-Generation Museum Tour-Guide Robot. In *Proceedings of the International Conference on Robotics and Automation*, vol. 3. IEEE.
- Thrun, S., & Mitchell, T. M. (1995). Lifelong Robot Learning. *Robotics and Autonomous Systems*, 15(1-2), 25–46.
- Topping, K. J. (2005). Trends in Peer Learning. *Educational Psychology*, 25(6), 631–645.
- United Nations' Department of Economic and Social Affairs (2017). *World Population Prospects: the 2017 Revision. Key Findings and Advance Tables*. United Nations Publications.
- Vanderborght, B., Simut, R., Saldien, J., Pop, C., Rusu, A. S., Pintea, S., Lefebvre, D., & David, D. O. (2012). Using the Social Robot Probo as a Social Story Telling Agent for Children With ASD. *Interaction Studies*, 13(3), 348–372.
- Verner, I. M., Polishuk, A., & Krayner, N. (2016). Science Class With RoboThespian: Using a Robot Teacher to Make Science Fun and Engage Students. *IEEE Robotics & Automation Magazine*, 23(2), 74–80.
- Wachter, S., Mittelstadt, B., & Floridi, L. (2017). Transparent, Explainable, and Accountable AI for Robotics. *Science Robotics*, 2(6).
- Wada, K., Shibata, T., Saito, T., Sakamoto, K., & Tanie, K. (2005). Psychological and Social Effects of One Year Robot Assisted Activity on Elderly People at a Health Service Facility for the Aged. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA)*, (pp. 2785–2790). IEEE.
- Wada, K., Shibata, T., Saito, T., & Tanie, K. (2004). Effects of Robot-Assisted Activity for Elderly People and Nurses at a Day Service Center. *Proceedings of the IEEE*, 92(11), 1780–1788.
- Wallbridge, C. D., Lemaignan, S., Senft, E., Edmunds, C., & Belpaeme, T. (2018). Spatial Referring Expressions in Child-Robot Interaction: Let's Be Ambiguous! In *Proceedings of the 4th Workshop on Robots for Learning (R4L) - Inclusive Learning, at HRI*.
- Waskom, M., Botvinnik, O., O'Kane, D., Hobson, P., Lukauskas, S., Gemperline, D. C., Augspurger, T., Halchenko, Y., Cole, J. B., Warmenhoven, J., de Ruiter, J., Pye, C., Hoyer, S., Vanderplas, J., Villalba, S., Kunter, G., Quintero, E., Bachant, P., Martin,

- M., Meyer, K., Miles, A., Ram, Y., Yarkoni, T., Williams, M. L., Evans, C., Fitzgerald, C., Brian, Fonnesbeck, C., Lee, A., & Qalieh, A. (2017). Mwaskom/seaborn: V0.8.1 (September 2017).
URL <https://doi.org/10.5281/zenodo.883859>
- Weintraub, J. (1997). The Theory and Politics of the Public/Private Distinction. *Public and Private in Thought and Practice: Perspectives on a Grand Dichotomy*, 1, 7.
- Wierwille, W. W., & Connor, S. A. (1983). Evaluation of 20 Workload Measures Using a Psychomotor Task in a Moving-Base Aircraft Simulator. *Human Factors*, 25(1), 1–16.
- Wills, P., Baxter, P., Kennedy, J., Senft, E., & Belpaeme, T. (2016). Socially Contingent Humanoid Robot Head Behaviour Results in Increased Charity Donations. In *Proceedings of the 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI) Extended Abstracts*, (pp. 533–534). IEEE.