Design of robot teaching assistants through multi-modal human-robot interactions.

Paola Ferrarelli, María T. Lázaro, and Luca Iocchi

DIAG, Sapienza University of Rome, Italy,
{ferrarelli, mtlazaro, iocchi}@diag.uniroma1.it

Abstract. The interest on introducing robots in schools has increased significantly in recent years. Robots in these environments are managed by educators who design teaching activities where the students can consolidate the knowledge acquired in the classroom by interacting with the robot.
In this context, the use of multiple modalities of communication can become a determining factor to achieve the success of the interaction and a better learning experience. However, the design of such multi-modal interactions can be a complex and time-consuming process, specially for teachers lacking of technical expertise.
In this paper, we propose a formalism for the description of multi-modal interactions based on the use of *interaction templates* which facilitates the design and management of the multi-modal behaviour by non-expert users (i.e., teachers). We provide an example of application of our approach on the educational context, using an autonomous robot as a teaching assistant, showing the usability and extendability of our system.

Keywords: Human-Robot Interaction; Multi-modal interaction, Interaction design, Learning with robots, Teaching with robots

1 Introduction

In the last years, the interest on introducing robotics in schools has increased significantly due to the possibilities it offers from the educational perspective. Robots in school become an interactive and more visually appealing educational tool attracting the interest of students who become active subjects during the learning process, instead of listening passively to a lesson. Through the use of robots in the classroom, the students can reinforce certain contents explained during the lessons in different ways, for example, by developing themselves real applications that can be executed on the robot or through interactive sessions with the robot. The latter is our application domain, for which we propose a framework for the generation of multi-modal interfaces for human-robot interaction.

In this context, the key actors of the educational process, (i.e., the teachers and the students) are usually non expert in the usage of the robotic tools available. As a consequence, the robotic tools to be used at school must have certain

desirable features, such as easy to be used by the teachers in order to design and prepare the lesson with the robot, and intuitive and easy to understand by the children. This usability can be increased by using multi-modal human-robot interactions (by combining for example, speech, graphical user interfaces, robot motion, etc.). However, when a multi-modal interaction is desired, the difficulty in designing and developing the interactions significantly increases.

This paper contributes to an easy definition and development of many short-term multi-modal interactions through the definition of a language for defining *interaction templates*. More specifically, a high-level formalism is defined to describe interactions in terms of the flow and of the content and the modalities of the basic communication actions. The interactions are described at two levels: i) a high-level description that provides an abstraction with respect to particular subjects and to the actual modalities, phrases, language, etc. used during the interaction (thus called *interaction templates)*, ii) a formal declarative description of the specific actions that are included in the interaction templates. The main advantages in using a formal language for specifying interaction behaviors are: *compactness*, notwithstanding the ability of representing multiple and complex interactions, *easy-of-use*, because it does not require knowledge about the internal implementations, *portability*, because of the abstraction with respect to the specific device, and *maintainability*, since such compact representation with a clear semantics allows for an easy management, updating, debugging, etc. The described method has been fully implemented and in this paper we propose a use case of using our framework to design and implement a school lesson where an autonomous robot supports the teacher during the explanation of a subject.

2 Related Work

Recently, the use of robots in classrooms, supporting teaching and education, spans from kindergarten to elementary and middle schools. A systematic review undertaken on published literature in seven international online bibliographic databases over a period of 10 years was done by Benitti [1]. She found, among others, the following interesting conclusions: the robot is used to teach technical subjects, such as robot assembling and programming; few experiments were done with home-made robots, while most were using Lego robotic kits; finally, most of the robotics activities were done during summer camp and/or after-school programs. The last data was confirmed also by Mubin et al. [9], that made a review of the applicability of robots in education. They remark that *efforts must be devoted not only to the development of robotic hardware and software for education but also to the design of learning material and appropriate curriculum and to the role of the teacher, that is directly linked to the role the robot plays in the learning activity*. Moreover, in the conclusion, they emphasize that the role of the robot is not to replace human teachers but being a stimulating, engaging and instructive tool, through which the interest of the students versus the curricula subject can be increased [8]. As experimented by Walker and Burleson [12] students seek activities that provide them with an appropriate level of challenge,

feelings of discovery, opportunity for physicality, and a sense of responsibility for the robot. About student motivation, Kanda [5] studied the effects of the use of social behaviors in a experiment with children taught uniquely by a Robovie robot using a learner-centered approach. Although children showed better social acceptance and motivation in the first two lessons, this effect disappeared after the novelty passed. This suggests the importance of the role of the teacher inside the classroom, in particular influencing how students accept new tools, as observed by Hussain [4] and Lindh [7]. However, as reported by the Teaching Profession in Europe [3], there still exists a technology gap in teaching. This report states that European teachers express moderate and high professional development need levels, especially in relation to 'Information and Communication Technology (ICT) skills for teaching' (57%) and 'new technologies in the workplace' (53%) topics (2013 data). These needs seem to be experienced almost uniformly, regardless of the subject they teach. This lack of technology skills could represent an obstacle to the implementation of the lessons using new technologies tools, like robots are, transmitting, as a consequence, negative attitude to the students towards them [11]. High-level formalisms have been used in contexts such as Human-Computer Interaction (HCI) [2], [10] but not in the educational robotics context. The use of a high-level formalism can reduce the efforts required to describe the human-robot interactions where the use of low-level formalisms could be hard to learn by teachers. We propose our methodology as a bridge between robots in education and high-level formalisms for multi-modal Human-Robot Interactions (HRI).

3 Methodology

We propose the following methodology to design school lessons in which an autonomous social robot acts as teaching assistant. In general, we propose to divide the lesson in n time slots in which we specify the actions that are realized by teacher, robot and students involved in the lesson. These actions can be arranged in a table as shown in Table 1.

We think the teacher should have an active role in supporting and managing the student's learning experience. Typical teacher actions A_{T_i} include: managing the time schedule, asking the robot to introduce itself and starting the test sessions, managing the student-student and student-robot interaction. The robot, like a real teacher assistant, does actions A_{R_i} that include: speaking to the classroom, calling the students, moving toward them, asking questions, displaying images, videos or statistical data. Finally, students actions A_{S_i} include: listening to the robot, answering to the questions, discussing between them and with the teacher, asking the robot about more explanations.

While it is sufficient to describe the actions performed by the teacher and the students in natural language, the robot actions A_{R_i} must be described in a formal way, since they must be executed by the robot software. The use of such a formalism provides a level of abstraction that hides the technical details to non-expert users and facilitates its posterior implementation and execution.

Time slot	Teacher	Robot	Student
1	A_{T_1}	A_{R_1}	A_{S_1}
$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	A_{T_n}	A_{R_n}	A_{S_n}

Table 1: Organization of a lesson with a robot teaching assistant.

4 Formalism

The formalism proposed in this paper to describe HRIs focus on interactions in which the robot takes the initiative of formulating questions and the answers are given by the users. These types of interactions are applicable to a variety of short-term interactions to provide information based on user choices or quizzes. Other forms of interaction are under investigation and will be considered in a future work.

Our approach for the generation of HRIs is based on the definition of *interaction templates*. As in the case of the templates existing in programming languages, the interaction templates allow to describe generic interactions that can be later *instantiated* according to a concrete situation.

Let us formally define an interaction template $\mathcal{T} = \langle \sigma_1, \ldots, \sigma_n \rangle$ as a sequence of actions $\sigma_i, i \in 1...n$ with

$$\sigma_i = \begin{cases} \emptyset\text{: no action} \\ a_i\text{: robot action} \mid \text{interaction } (\mathit{state}, \mathit{ask}, \mathit{answer}) \\ \phi\,?\,\mathcal{T}_1 : \mathcal{T}_2 \\ \text{choice}(\mathcal{T}_1, \ldots, \mathcal{T}_n) \\ \texttt{LABEL} \\ \texttt{GOTO LABEL} \end{cases}$$

The conditional operation $(\phi\,?\,\mathcal{T}_1 : \mathcal{T}_2)$ allows to enable different interaction templates during the course of the interaction. The boolean expression ϕ is formulated over a set of variables whose meaning may comprise the result of both robotic or interaction routines. The syntax of this operator is the following: if ϕ evaluates as TRUE, the interaction template $\mathcal{T}_1$ is activated, otherwise $\mathcal{T}_2$ is executed. Similarly, the choice operation allows to select one template among a set $(\mathcal{T}_1, \ldots, \mathcal{T}_n)$ based on a random, predefined or learnt policy. Finally, `LABEL` and `GOTO LABEL` allows to annotate particular actions and introduce loops in the interaction. Notice these operators produce an interaction graph which makes the formalism suitable for generating an equivalent Petri-Net Plan (PNP) [14] to manage the interaction, as explained in the next section.

In contrast to HCIs, the flow of an HRI includes both *robotic* (e.g., motion and perception) and basic communication actions. Our template-based interaction design considers three main interaction actions - *state*, *ask*, *answer* - whose syntax is defined below:

– *State* actions are used by the robot to communicate some fact to the user.

$$state < \texttt{Type}, P_1, \ldots, P_k >$$

where `Type` is the type of the state action and $P_1, \ldots, P_k$ are the (possibly empty) instantiated parameters (i.e., ground values needed to instantiate the statement).

– *Ask* actions are used to denote questions from the robot to the user. They have a similar structure to the *state* action and, in addition, require to specify a set of possible answers S.

$$ask < \texttt{Type}, P_1, \ldots, P_k, S >$$

– *Answer* actions represent answers given by the user and always refer to the last *ask* action in the template.

$$answer < X >$$

where X is the name of a variable that will be instantiated with a value corresponding to the choice of the user. After the user interaction, the variable X will be assigned to one of the values specified in the set S related to the last *ask* action.

The set of possible answers S to a given question is predefined. This design choice allows for increasing robustness of recognition and correctness of interaction, since allowing for a completely open set of answers is still not manageable by current technologies. This set S can be specified by the user as a constant set (e.g., $\{yes, no\}$) or obtained by an external procedure accessing a database, a knowledge base, on-line resources, etc.

So far, this formalism provides a high-level description of the interaction. In general, the actual execution of an action will depend on the specific task and modality of interaction chosen. Our approach allows for a multi-modal definition of each action. Modalities can vary depending on the application, the available interaction devices on the robot, etc. For example, we can consider the case of a robot equipped with a touch-screen in which text, images and videos can be displayed and with a Text-to-Speech (TTS) system to transmit messages to the user by voice. Input from the user can be received through the touch-screen or an Automatic Speech Recognition (ASR) system.

In order to achieve a multi-modal interaction, we implement for each action `Type` the modalities γ that we want to make available (e.g., $\gamma = \{text, image, TTS, video, ...\}$). Let us define $\varphi_\gamma = \texttt{Type\_}\gamma(P_1, \ldots, P_k)$ a function that given an action `Type`, its parameters $P_1, \ldots, P_k$ and the selected modalities γ returns the actual interaction φ_γ to be executed. Let us also denote $\varphi = \bigcup_\gamma \varphi_\gamma$ the set of actual interactions over all modalities available (e.g., $\varphi = \{T, I\}$ contains a text and image when modalities $\gamma = \{text, image\}$ are available). The set of interaction actions are specified as:

$state < \texttt{Type}, P_1, \ldots, P_k >$	$ask < \texttt{Type}, P_1, \ldots, P_k, S >$	$answer < X >$
$\varphi_\gamma = \texttt{Type\_}\gamma(P_1, \ldots, P_k)$ $comm\_output(\varphi)$	$\varphi_\gamma = \texttt{Type\_}\gamma(P_1, \ldots, P_k)$ $comm\_output(\varphi)$ $comm\_prepare(S)$	$X := comm\_input()$

Here, $comm\_output(\varphi)$ represents the system-specific procedures to be carried out to finally communicate the interaction to the user using the chosen modalities, for example, the display of an image I on a GUI or the transmission of a selected text TTS to the speech component. $comm\_prepare(S)$ represents the process in which the robot prepares the interaction devices for an answer from the user among the set of possible answers S. Finally, *comm_input* assigns the result of the interaction to the given variable.

The presented approach is easy to manage and requires little effort from the designer. In practice, s/he has only to provide: 1) the interaction template $\mathcal{T}$ and 2) the description of the interaction actions $\texttt{Type\_}\gamma(.)$. Section 6 will provide examples of implementation and execution of this formalism.

5 On-line execution of the interaction template

On-line execution of the interactions is based on three main components. The first one is the Petri Net Plan (PNP) engine that executes the interaction plan according to its semantics. The PNP execution algorithm is described in details in [14] and implemented in the PNP library. The second component is the PNP-ROS bridge (also available in the PNP library) that allows the execution of PNPs on the robot. The third component is the implementation of the set of robotic and interaction actions described in the PNP. In this paper we focus on the interaction actions which are implemented in a client-server fashion, where the robot actions are clients that exchange commands and results of the interactions with an interaction server. The client-server architecture also allows for distributed computation and portability. In our implementation, we use a Linux laptop for the control of the robot and the execution of the PNP and a Microsoft Windows tablet for user interaction.

While most of these components are already available, the contribution described in this paper is related to the instantiation and the execution of the interaction actions described in the previous section. This is obtained with the implementation of a Multi-Modal User Interface (MMUI) that manages a Python GUI and a C# speech server using the multi-language Microsoft Speech Recognition and Synthesis engine.

More specifically, the MMUI component acts as a server, executes the interaction actions when enabled by the PNP engine, and returns the results of the interactions (i.e., input from humans to the robot) as conditions that are evaluated by the PNP engine to enable the proper transitions. The MMUI component implements the actions *state*, *ask*, and *answer* by first collecting all the modalities for producing an interaction. In particular, *state* implements the *comm_output* function (i.e., communication from the robot to the human) by

showing a statement with one or more of the output modalities available on the robot (text, images or videos shown on the tablet, spoken sentences by the robot). *ask* implements the *comm_output* function in the same way as in *state* and then it implements the *comm_prepare* function for receiving the input from the user (e.g., by displaying buttons on the GUI and loading specific speech recognition grammars). Finally, *answer* implements the *comm_input* function that assigns the result of the interaction (either through GUI buttons or speech) to the given variable.

During the execution of the functions `Type_`$\gamma(.)$ defined in the interaction actions, variables $P_1, \ldots, P_n$ are always instantiated and the action execution algorithm guarantees the storage of the values of the variables and the use of the corresponding values when needed.

6 Implementation and Experiment

As already mentioned, the proposed framework has been fully implemented and tested by using the social robot Diago interacting with students in a classroom helping the teacher to do a Physics lesson[1]. Diago is a social mobile robot used for experiments in social human-robot interaction, knowledge representation and reasoning, and cognitive robotics. Diago stands 170 cm tall, is built on top of a Segway mobile base, equipped with a RoboTorso that contains laser sensors for motion, RGBD camera, microphone, and audio speakers for HRI, a laptop for controlling the mobile platform and the sensors, and a tablet for HRI and speech recognition and synthesis. We propose the design of a lesson of 50 minutes about Gravity, organized like in Table 2.

The students and the teacher are standing up in a classroom to facilitate Diago movements and its interaction with students. The robot moves around approaching the students with a welcoming message. The interaction template and the definition of the interaction actions are defined below:

$\langle approach, state <$ `Welcome`, "$Physics$" $>, state <$ `InfoArgument`, "$Gravity$" $>\rangle$

```
Welcome_TTS(P1):
     Hello everybody. It's time for a @P1 lesson.

InfoArgument_TTS(P1):
   choice:
      Today, we will learn about @P1.
      [P1=Gravity] Together we will discover why objects are heavy.
      [P1=Force] Together we will discover why objects move.

Welcome_text(P1):
   choice:
```

[1] More information about the robot used and the results of the experiments are provided in the web site https://sites.google.com/a/dis.uniroma1.it/robot-at-school/.

Time Slot(min)	Teacher Actions	Robot Actions	Student Actions
Introduction, 5	Ask the robot to introduce the argument of the lesson.	Move inside the circle.	Arrangement in a circle.
Pre-test, 10	Ask the robot to start the pre-test session. Manage the student-robot interaction.	Call students. Move toward him/her. Tell the first question. Get the answer. Tell the next one. Repeat for each student.	Answer to all the questions.
Discussion, 10	Start and Stop the discussion activity.	Display statistic of answers, videos and images.	Free discussion.
Ask to Diago, 5	Manage the students.	Answer to students.	Ask to Diago for deepening.
Post-test, 10	Ask the robot to start the post-test session. Manage the student-robot interaction.	Call students. Move toward him/her. Tell the first question. Evaluate it: if correct, tell the next one; if wrong, repeat the question. Repeat for each student.	Answer to all the questions.
Conclusion, 10	Ask the robot to summarize the lesson.	Summarize the lesson. Thank the class. Make congratulations. Tell the next argument. Ask to arrange at the desk. Say goodbye message	Greet and thank to teacher and Diago. Arrangement at the desk.

Table 2: Lesson plan with a robot assistant.

```
    Welcome to the @P1 lesson.
    Good morning.

InfoArgument_text(P1):
  choice:
    @P1 is the topic we are going to study today.
    Argument: @P1. Lesson Schedule: introduction, pre-test, discussion,
              Q&A, post-test, summary
```

In order to describe the interaction actions we use the following syntax: 1) a `choice` operator is presented to select one of the possible actions (it can be omitted if there is only one option), 2) `@P1` is replaced by the exact content of the variable P_1. Thus, for `Welcome_text(`"*Physics*"`)` action, sentences "`Welcome to the` *Physics* `lesson?`" or "`Good morning`" would be generated, 3) it is possible to particularize a concrete action given a specific parameter value using the syntax `[P1=ParameterValue]`. For example, in `InfoArgument_TTS(`"*Gravity*"`)`, both "`Together we will discover why objects are heavy.`" or "`Today, we will learn about Gravity`" could be selected.

An example of execution of this template, together with the modalities activated is shown below:

R : [TTS] *Hello everybody. It's time for a Physics lesson.*
R : [text] *Good morning.*
R : [TTS] *Today, we will learn about Gravity.*
R : [text] *Argument: Gravity. Lesson Schedule: introduction, pre-test, discussion, Q&A, post-test, summary*

Then, the teacher proposes a pre-test of 4-5 questions to students. So, the robot calls each student by name, goes forward him/her and ask him/her for questions like:

"If an object is brought to the Moon, its mass:
a) decreases b) disappears c) increases d) remains the same"

These type of questions follow a template of the form $\langle ask < \texttt{Quiz}, \texttt{GetChoices}() >, answer < Result > \rangle$, where the questions can be stored in a database containing also multimedia data. `GetChoices`() is a generic function that queries the designed database to retrieve the set of possible answers for each of the proposed questions of this example, and *Result* is a variable that will store the selection made by the student, which will contain the semantic meaning of the answer, i.e., if the answer was correct or wrong. Figure 1 shows an example of a question formulated to a student using our Graphical User Interface.

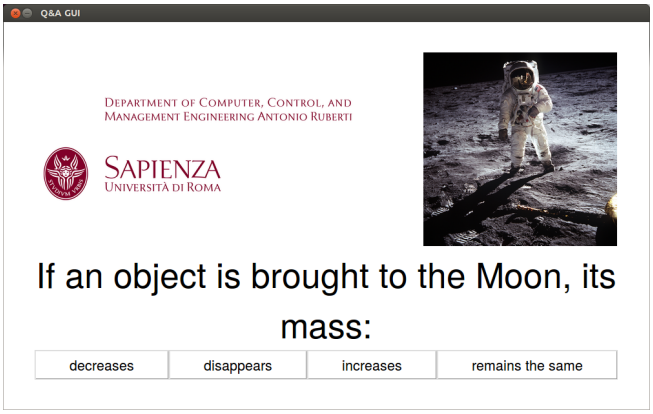

Fig. 1: Example of GUI interaction with a student in a lesson about Gravity.

Here, the student can answer either by using the touch-screen on the robot or by voice. Once the selected answer is registered or recognized by the robot, the next question is displayed. During the pre-test session the correct answer is not displayed because the aim is to catch students misconceptions about the argument of the lesson (Gravity in this example). When all the students have answered the questions, the teacher asks the robot to display the statistics of correct/wrong answers, starting the discussion about the wrong ones. The robot assists the teacher by showing videos or images (e.g., the moon landing image) in order to create cognitive conflict in the group, stimulating in this way the discussion and the recognition of the correct answer by the group. Following, a question and answer (Q&A) activity will give to students the opportunity to interact with Diago, asking for deepening about Gravity, using images and videos. The post-test session is similar to the pre-test one, because the robot

calls each student by name and goes to her/him asking for the same questions of the pre-test, collecting the answers but, unlike before, if the answer is wrong a failure message, an invitation to try again and the same image/video displayed during the discussion session are shown. A feasible interaction template for a post-test session and an example of interaction is shown below:

$$\langle \texttt{LABEL1}, ask < \texttt{Quiz1}, \texttt{GetChoices}() >, answer < Result1 >,$$
$$Result1 = wrong?state < \texttt{Answer}, \text{“}wrong\text{”} >, \texttt{GOTO LABEL1}$$
$$: state < \texttt{Answer}, \text{“}right\text{”} >, \texttt{LABEL2}, ask < \texttt{Quiz2}, \texttt{GetChoices}() >, ... \rangle$$

```
Answer_text(P1):
   choice:
      The answer is @P1.
      [P1=right] Congratulations!
      [P1=wrong] I am sorry, this is not the right answer. Try it again.
Answer_image(P1):
   choice:
      [P1=right] img_happy_smile.png
      [P1=wrong] img_sad_smile.png
```

R : [Both TTS and text] *If an object is brought to the Moon, what happens to its mass?*
R : [Image](See Fig. 1.)
H : [Speech] *disappears*
R : [Both TTS and text] *I am sorry, this is not the right answer. Try it again.*
R : [Image] (a smile or sad emoticon)
Due to the `GOTO` instruction, question is reformulated until the correct answer is obtained.

Then, the interaction continues with other quizzes.

Finally, Diago summarizes what they learn during the lesson and greets the classroom. The role of the teacher, during the lesson, is managing the student-robot interaction, for example checking that each student start and stop his/her own interaction, without interrupting it; managing the time in order to maintain the schedule; managing the student-student interaction, a social activity that high school students needs and appreciate but rarely they are able to control it. Finally, the robot asks the students to sit down at their own desk and greets them with a goodbye message.

R : [TTS and text] *Goodbye! Have a nice day!*

6.1 Experimental validation

A reduced instance of the lesson described above has been used for a user study. During an Open event in our Department, we hosted 50 high school students of around 18 years old (10-12th grade).

Interaction with the robot during the event	Yes	No	
Perceived Intelligence	**average-variance**	**average-variance**	**P value**
Incompetent versus Competent	4.30-0.47	3.91-0.54	0.40
Ignorant versus Knowledgeable	4.41-0.63	4.22-0.56	0.38
Irresponsible versus Responsible	4.00-0.75	4.09-0.67	0.38
Unintelligent versus Intelligent	4.35-0.62	4.09-0.93	0.20
Foolish versus Sensible	3.23-1.07	3.50-0.58	0.07
Perceived Safety			
Anxious versus Relaxed	4.06-1.68	4.56-0.45	**0.0008**
Calm versus Agitated	1.64-0.99	1.65-1.07	0.45
Quiescent versus Surprised	3.71-0.85	3.78-1.14	0.26

Table 3: Godspeed questionnaire data about perceived intelligence and safety. The full results are in the web site mentioned at the beginning of this section.

We proposed them short Physics lessons (15 minutes), organized as described previously. After the lesson we distribute a questionnaire Godspeed Questionnaire Series (GQS) [13] that is frequently used for HRI evaluation. We used GQS for assessing the success of the robot, evaluating if the emotional state and the impression of the robot was influenced by the fact that the students were interacting or not with it. A significant sample of data analysis of perceived intelligence and safety is showed in Table 3.

The general scores were all positive, showing a general acceptance of the experience. The relatively high P-values calculated for almost all the GQS parameters show that, during the curricula lesson with a robot, students' emotional state and their impression of the robot was not influenced by the fact that they interacted with it. A notable exception was found for the anxiety state, for which the data analysis shows a significant higher variance in the students that interacted with the robot. This fact indicates that interaction with a robot during a teaching experience can generate anxiety that must be taken into account by the teachers. Nonetheless, the overall result confirms that the role of the robot as teacher assistant is well accepted without the need that each student has to interact with it.

7 Conclusions

In this paper, we have presented a formalism to describe templates of multimodal interactions. Once the interaction template has been designed, it is instantiated and executed on the robot by using the PNP formalism.

The use of the templates is intuitive and facilitates the work of the designer, who must not necessarily be an expert from the technological point of view, and, as shown in the example, the templates are easily extendable and re-usable in different contexts. A user-friendly mechanism to populate the robot database represents a good opportunity to facilitate the use of such tools by the teacher, being a chance to enhance his/her technology skills while trying to catch and hold the student interest for the subject.

Work in progress includes the application of this methodology in a more structured teaching experience and a deeper evaluation of the results. Another

interesting direction is personalization of the interaction that allows to maintain the interest over time of the students, as demonstrated by Lee [6].

Acknowledgements

This work has been partially developed within the COACHES project. COACHES is funded within the CHIST-ERA 4^{th} Call for Research projects, 2013, Adaptive Machines in Complex Environments (AMCE) Section. Sapienza University is funded by MIUR (Italy).

References

1. Fabiane Barreto Vavassori Benitti. Exploring the educational potential of robotics in schools: A systematic review. *Computers & Education*, 58(3):978 – 988, 2012.
2. P. Bottoni, M. F. Costabile, and P. Mussio. Specification and dialogue control of visual interaction through visual rewriting systems. *ACM Trans. Program. Lang. Syst.*, 21(6):1077–1136, November 1999.
3. European Commission/EACEA/Eurydice. The teaching profession in Europe: Practises, perceptions, and policies (eurydice report). Technical report, Luxembourg: Publications Office of the European Union., 2015.
4. S. Hussain, J. Lindh, and G. Shukur. The effect of lego training on pupils school performance in mathematics, problem solving ability and attitude: Swedish data. *Journal of Educational Technology and Society*, 9(3):182194, 2006.
5. Takayuki Kanda, Michihiro Shimada, and Satoshi Koizumi. Children learning with a social robot. In *Proc. of the Seventh Annual ACM/IEEE Int. Conf. on Human-Robot Interaction*, HRI '12, pages 351–358, New York, NY, USA, 2012.
6. Min Kyung Lee, Jodi Forlizzi, Sara Kiesler, Paul Rybski, John Antanitis, and Sarun Savetsila. Personalization in hri: A longitudinal field experiment. In *7th ACM/IEEE International Conference on Human-Robot Interaction*, March 2012.
7. Jörgen Lindh and Thomas Holgersson. Does lego training stimulate pupils' ability to solve logical problems? *Comput. Educ.*, 49(4):1097–1111, December 2007.
8. Ruben Mitnik, Miguel Nussbaum, and Alvaro Soto. An autonomous educational mobile robot mediator. *Autonomous Robots*, 25(4):367–382, 2008.
9. O. Mubin, C. Stevens, S. Shahid, A. Al Mahmud, and J.-J. Dong. A review of the applicability of robots in education. *Technology for Education and Learning*, 2013.
10. Robbie Schaefer, Steffen Bleul, and Wolfgang Mueller. *Dialog Modeling for Multiple Devices and Multiple Interaction Modalities*, pages 39–53. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.
11. A. M. Vollstedt, M. Robinson, and E. Wang. Using robotics to enhance science, technology, engineering, and mathematics curricula. In *American Society for Engineering Education Pacific Southwest annual conference*, 2007.
12. Erin Walker and Winslow Burleson. *User-centered design of a teachable robot*, volume 7315 of *Lecture Notes in Computer Science*, pages 243–249. 2012.
13. A. Weiss and C. Bartneck. Meta analysis of the usage of the godspeed questionnaire series. In *2015 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 381–388, Aug 2015.
14. V. A. Ziparo, L. Iocchi, P. U. Lima, D. Nardi, and P. F. Palamara. Petri net plans - A framework for collaboration and coordination in multi-robot systems. *Autonomous Agents and Multi-Agent Systems*, 23(3):344–383, 2011.