Playte, a Tangible Interface for Engaging Human-Robot Interaction

David Johan Christensen, Rune Fogh and Henrik Hautop Lund

***Abstract*— This paper describes a tangible interface, *Playte*, designed for children animating interactive robots. The system supports physical manipulation of behaviors represented by LEGO bricks and allows the user to record and train their own new behaviors. Our objective is to explore several modes of interaction, i.e. direct remote control, tangible programming, programming by demonstration, and programming by training, to learn the design principles for more accessible, engaging, and playful robots. We evaluate the system experimentally and report on key observations from play sessions. We conclude that *Playte* facilitates playful activities and is appropriate for the intended target group (age 6+). Further, we discuss lessons learned regarding pros and cons of the different supported interactions modes.**

I. Introduction

Children can build their own personal robots based on modular reconfigurable robots and robotic construction kits by direct and intuitive assembly of robotic components. The same cannot be said about programming behaviors for the robot, which often requires sophisticated programming skills, making it inaccessible and unattractive for many children. We approach this challenge by providing the children with a choice of different programming paradigms, such as visual programming, tangible programming, programming by building, programming by demonstration and programming by training. We anticipate that such an approach has the potential to make the creation of behaviors more accessible, engaging, and playful even for young children.

In this paper we describe a tangible interface, called *Playte* (an arbitration of “play plate”). It is designed to explore playful physical programming of modular robotic playware. Our objective is to develop an intuitive interface for children to explore the creation of behaviors for interactive robots, see Fig. 1. The approach of *Playte* is to enable direct physical manipulation of a robot’s behavior as well as the possibility to create new behaviors without the need for a conventional computer interface. To realize this objective we let LEGO bricks represent preprogrammed behaviors which is activated in the robot when placed on the *Playte*. By utilizing a subsumption architecture the robot can run several behaviors simultaneously. Gamepad control enables the child to create new behaviors by either recording a sequence of actions or training new behaviors with a machine learning method. These new behaviors is also represented with a LEGO brick and can therefore be used in an equivalent fashion as the preprogrammed behaviors.

D. J. Christensen, R. Fogh, and H. H. Lund is with the Center for Playware, Department of Electrical Engineering, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark {djchr, rufo, hhl}@elektro.dtu.dk

Fig. 1. Two children, age 6 and 7, interacting with a robot. The children have used the *Playte* tangible interface to give the robot a certain behavior.

In the rest of this paper we first review related work in Section II. In Section III we describe the design of *Playte*, the design of preprogrammed behaviors, and the control strategies for subsumption arbitration, recording and training new behaviors. The robotic platform used for evaluation is described in Section V. In Section VI we present example experimental trails to illustrate the functionality of the *Playte* interfaced to the robotic platform and present key observations from play session with children from the intended target group (aged 6 to 11). In Section VII we discuss the pros and cons of the different interaction modes and in Section VIII we give conclusions and describe future work.

II. Related Work

In this paper we consider tangible programming of robotic playware, which can be modular robotic systems [23] or robotic construction kits, designed to enable a non-expert user to construct robotic artifacts for playful activities [12], [17], [20], [21]. We develop technologies for playful activities, because of play’s positive impact on the child development, as observed by Vygotsky [25]:

> *In play a child always behaves beyond his average age, above his daily behavior. In play it is as though he were a head taller than himself.*

In addition, play and games can motivates us to do tedious tasks [4], rehabilitation exercises [13] and learning activities [19]. What motivates us to enter the state of play is a matter of subjective and psychological preferences. The PlayGrid [5] is a model which attempt to encapsulate these differences to assist designers when developing objects for play.

Systems for construction of robotic creatures utilize different strategies to enable users to control the behavior of

Fig. 2. The *Playte* is a tangible interface for easy assessible behaviors-based robot control. Shown are also preprogrammed behavior-bricks and trainable bricks, as well as a gamepad used for training/recording new behaviors and for direct remote control.

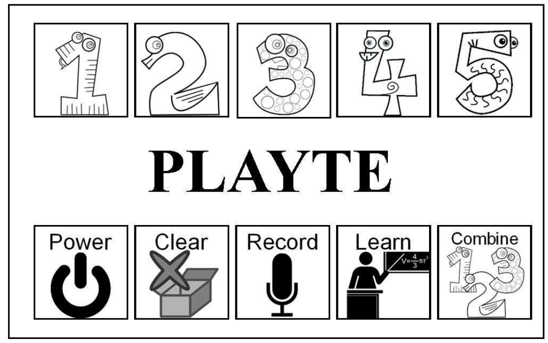


Fig. 3. An overview of the *Playte* tangible interface. Each icon corresponds to a slot for attaching special LEGO bricks. Bricks attached to the top row are active behaviors on the robot. The bottom row contains on/off, deleting a behavior, record a behavior, train a behavor, and combine several behaviors from the top row into a new one.

their creation: The LEGO Mindstorms, LEGO WeDo, and PicoCricket [20] all support visual programming, the Topobo system lets the user record and playback motor sequences by directly interacting with the system [17], roBlocks (now Cubelets) utilize a physical-programming strategy where the behavior is emerging from the modules interactions with each other and their environment [21]. In this paper we report on the design of an interface for modular playware which, like some of these systems, support remote control, tangible behavior-based control, learning by demonstration and learning by training. A main contribution of this paper is to demonstrate a single system where all these different interaction modes are integrated. Thereby, the different modes of interaction can in future work be evaluated and compared more directly.

Modular playware and tangible interfaces [9], [22] share some characteristics since they both enable users to manipulate data directly as physical objects to interact with computers in novel ways. Related examples of tangible interfaces include AlgoBlocks [24], Tangible Programming Brick [14], electronics blocks [27] and Algorithmic Bricks [11]. Research in tangible interfaces for children indicates that the interaction is faster, easier and increase the children's exploratory approach to problem solving, when compared with a mouse-based-GUI interface for making jigsaw puzzles [1]. Tangible user interfaces are also shown to be a more attractive mean of interacting than its graphical counterpart [28], especially for girls [8].

III. Design of the Playte

The *Playte* is a tangible interface. It has a square outer shell build mainly from LEGO Systems bricks, see Fig. 2. On the *Playte's* lid there are ten slots where the user can place behavior bricks, see illustration in Fig. 3. A slot consists of a special LEGO brick with electrical connections and a LED diode for user feedback. The top row of five slots selects which behaviors are active in the robot. The last row of five slots allows the user to delete a behavior, record a behavior, train a behavior, and combine several behaviors into one. Behaviors are represented by special LEGO bricks, which are modified to contain a resistor for identification. The electrical resistances are read using AD converters by a microcontroller inside the shell of the *Playte*. The microcontroller periodically sends the state of the *Playte* slots to a PC using serial communication. The PC forwards the *Playtes* state to the robot together with any user input from a gamepad. The *Playte* board state is a 10-tuple containing the board's slots and the corresponding IDs of attached behavior bricks (if present):

$$BoardState = \langle (slot_0, brick_{id}), ..., (slot_9, brick_{id}) \rangle \quad (1)$$

Based on this state the selected behaviors, record/playback, training, and combining are executed directly on the robot's microcontroller. Note that the PC acts as a simple communication relay/gamepad interface/data logger/sound playback and could be removed in a future work.

A. Tangible Interaction

A behavior brick is a modified LEGO brick which represents a simple preprogrammed robot behavior such as "Dance", "Hide", and "Parrot Song". We have written the behaviors to be largely independent on the particular morphology of the robot, so that the same behaviors potentially can be used on different types of robots that the users may eventually build themselves. Alternative, in future work, children could combine the tangible programming presented here, with visual programming language, such as Scratch [18], to create their own behaviors.

We consider a robot with m actuators and n sensors, which can be selected on runtime by the user. A behavior program, $behavior_{id}(S, M) \rightarrow \mathbb{R}^m$, is a simple function running on the robot, which takes as input the robot's current sensory state, $S \in \mathbb{R}^n$, and the current set points for the actuators, $M \in \mathbb{R}^m$, and returns a new tuple, potentially unchanged, containing the actuators set points. The behavior program is activated when its corresponding behavior brick, $brick_{id}$, is placed on the board.

Algorithm 1 Subsumption(BoardState)

$M = \langle 0, 0, 0, \ldots, 0\rangle$
$S = \langle s_0, s_1, s_2, \ldots, s_{n-1}\rangle$
for $i = slot_0$ to $slot_4$ **do**
  if $brick_{id}$ is on $slot_i$ **then**
    $M = behavior_{id}(S, M)$
  end if
end for
Control motors based on M

Algorithm 2 Trained(S, M)

Sort $TrainEx$ based on $\|S - S_i\|$, $S_i \in TrainEx$
$kNearest$ = first k elements of $TrainEx$
$M_{new} = \frac{1}{k}\sum_{i=1}^{k} M_i$, $M_i \in kNearest$
return M_{new}

The *Playte* has five slots for behavior bricks to allow the children to use more than one behavior at a time. Thus we need a mechanism for arbitration of several active behaviors. For this purpose we utilize a simplified subsumption architecture [2], which we expect will be intuitive to the user because of its simplicity and effectiveness. The five slots are mapped to five layers in a subsumption architecture with the left-most slot corresponding to the lowest layer. Behaviors at a higher layer in the subsumption architecture can inhibit the control output of the behaviors at lower layers. The utilized subsumption architecture based strategy is illustrated in Algorithm 1. In addition to the motor output all active behaviors may also trigger sound effect playback on the PC.

B. Recording Behaviors

A special brick allow the user to record a new behavior brick. The use case for recording a behavior is as follows: First the user place a programmable brick on the "Record" slot, and then use the gamepad to control the actuators of the robot for a period of time. The robot's microcontroller periodically samples the state of the actuators, M_i, at a specific time, t_i, since the recording was started:

$$Recording = \langle (t_0, M_0), (t_1, M_1), ..., (t_N, M_N)\rangle \quad (2)$$

The recording is complete when the brick is removed from the "Record" slot or when a specific amount of memory has been used (here equivalent to 7.5 seconds). Later, the recorded behavior can be played back in a loop when placed on a slot on the top row of the *Playte* or it can be deleted by placing it on a specific slot.

C. Training Behaviors

The user have two ways to train new behaviors based on teaching the robot: i) A direct reactive sensor-motor mapping or ii) A direct sensor-behavior mapping.

Training a Reactive Controller: To train a new reactive behavior the user places a programmable brick at the "Train" slot. Now the user can teach the robot what to do in specific sensory conditions by demonstrating it using the gamepad to control its motors. The training stops when the brick is removed from the slot and the trained behavior brick can then later be activated by placing it on the top row of the *Playte*.

Algorithm 3 Combined(S, M)

Sort $CombEx$ based on $\|S - S_i\|$, $S_i \in CombEx$
$kNearest$ = first k elements of $CombEx$
for $i = slot_0$ to $slot_4$ **do**
  Select most frequent $behavior_{id}$ on $slot_i \in kNearest$
  if $behavior_{id} \neq null$ **then**
    $M = behavior_{id}(S, M)$
  end if
end for
return M

Training is implemented by utilizing a k-nearest neighbor algorithm (k-NN). When a behavior brick is trained the system samples training data as a mapping from the robot's sensor values, S_i to the motor set points, M_i:

$$TrainEx = \langle (S_0, M_0), (S_1, M_1), ..., (S_N, M_N)\rangle \quad (3)$$

In the case where the number of training samples has filled the amount of memory allocated on the robot (here equivalent to 50 unique samples), new training samples will overwrite a randomly selected old training sample.

When the trained behavior brick is made active on the *Playte* the system uses the k-NN algorithm to find the appropriate motor output in the given sensory situation, as illustrated in Algorithm 2.

Training by Combining Behaviors: The user can also combine several behaviors into a new behavior through a similar training process. A programmable brick is placed at the "Combine" slot while one or more behavior bricks are active on the top row slots of the *Playte*. The user then continuously selects which behaviors should currently be active by using the buttons on the gamepad (1-5) to select (pressed) or deselect (released) the corresponding slots.

The combining behavior strategy is also implemented using a k-NN algorithm. While training, the system periodically saves the training data, a data set at time step i will have the following form:

$$CombEx = \langle (S_0, Bricks_0), ..., (S_N, Bricks_N)\rangle \quad (4)$$

Where $Bricks_i$ is the state of the top board row as selected by the user in timestep i:

$$Bricks_i = \langle (slot_0, brick_{id}), ..., (slot_4, brick_{id})\rangle \quad (5)$$

The control strategy, illustrated in Algorithm 3, uses the k-NN algorithm to find k examples of behaviors the user has selected in a similar situation. For each behavior slot the most frequent behavior is selected and executed as part of the subsumption architecture.

IV. Design of Preprogrammed Behaviors

The preprogrammed behavior bricks are marked with an icon that illustrates the corresponding behavior to the child,

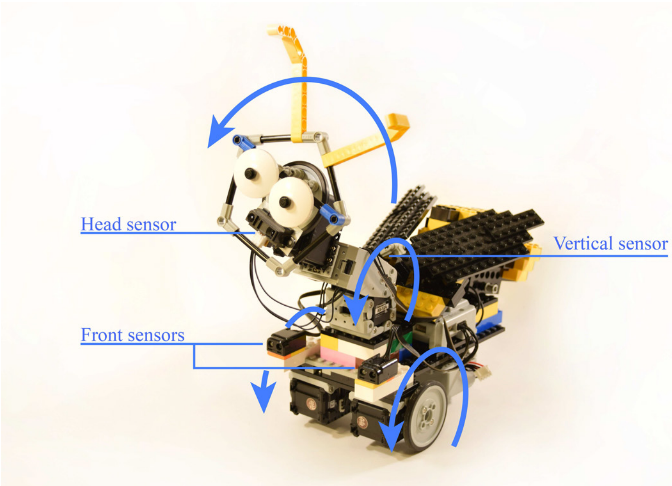


Fig. 4. The robotic platform used to test the *Playte* system. The robot is based on components from the Robotis Bioloid kit and LEGO bricks.

such as play, hide, and seek. The behaviors will cause the robot to express signs of emotions, through movements, sounds, music, and its response to interactions with the child and the environment. The idea is to exploit the child natural tendency to mimic and feel the emotions of others, in this case the emotions expressed by the robot. Such form of emotional contagion is a well know psychological phenomena [6], and our hypothesis is that such emotional contagion can be utilized as a play force to create a good play dynamics [10].

Table I provides an overview of the preprogrammed behaviors, what causes the behaviors to start and stop and a short description of their action. The start and stop conditions are based on simple sensory states and are shared by all behaviors. "Sound" is the presence of a loud noise such as a clap. "Nearby" is an activity of short range distance sensors above a certain threshold and "Visible" is a certain activity level on a long range distance sensors.

V. Design of Robotic Platform

To evaluate the functionality and usability of the *Playte* we have developed a simple robotic platform. The robot has the appearance of a bee, is mobile and has a movable head as well as several sensors as shown in Fig. 4. The moveable head enables the robot to express several emotional clues such as lowering its head as a scared ostrich or tilting its head as an attentive dog.

The robot is based on components from the Bioloid kit from Robotis combined with LEGO elements. The servos are Dynamixel AX-12A and the robot controller is a CM510, which contains an 8-bit Atmel Atmega2561 MCU with 8 KB SRAM and a clock frequency of 16 MHz. Wireless radio communication to the PC hosting the *Playte* is based on a ZigBee module. Low level firmware as well as high level control strategies and behaviors are implemented as part of the Assemble and Animate (ASE) control framework for modular robots [3].

The robot has two wheels and a two DOF movable head, a total of four DOF (see Fig. 4). The robot can move forward with a velocity of $V_{max} = 17cm/s$. Regarding sensors, the robot has a loud noise detecting microphone, three short range IR distance sensors ($1.5-15cm$ range), and one long range IR distance sensor ($10-80cm$ range). The three short distance range sensors are placed in the front (left and right) and on the back (pointing vertically upwards). The long range distance sensor is placed in the head of the robot pointing forward.

VI. Evaluation

This section presents typical interaction examples, an example training trial and key observations from play sessions.

A. Tangible Interaction

The objective of this example is to illustrate the interaction in a typical tangible programming scenario as well as exemplify the effect of the subsumption architecture utilized.

By placing the "Dance" behavior brick on the top row of the *playte* the robot will start to dance when a rhythmic sound signal, such a clapping, is produced by the child. Fig. 5(a) shows the sound input (claps) and motor output of this behavior. In order to generate the rhythmic and randomly varying dance movement based on sound input we utilize the dance control architecture of Keepon [15].

By placing several behavior bricks on the top row of the *playte* the subsumption architecture is utilized to arbitrate the behaviors. Fig. 5(b) shows an example interaction with the "Tickle Play" behavior brick placed on the lowest priority *Playte* slot and the "Attack" behavior brick on a higher priority slot. The robot behavior is controlled by one of the two behaviors depending on the sensory condition. The head sensor activates the "Visible" sensory start condition of "Attack" and the other three distance sensors triggers the "Nearby" sensory start condition of the "Tickle Play" behavior. In the cases where both sensory conditions are true, the higher priority behavior subsumes the behaviors with lower priority. In this example the "Attack" behavior subsumes the "Tickle Play" behavior around 4-5 seconds into the trial.

B. Training Behaviors

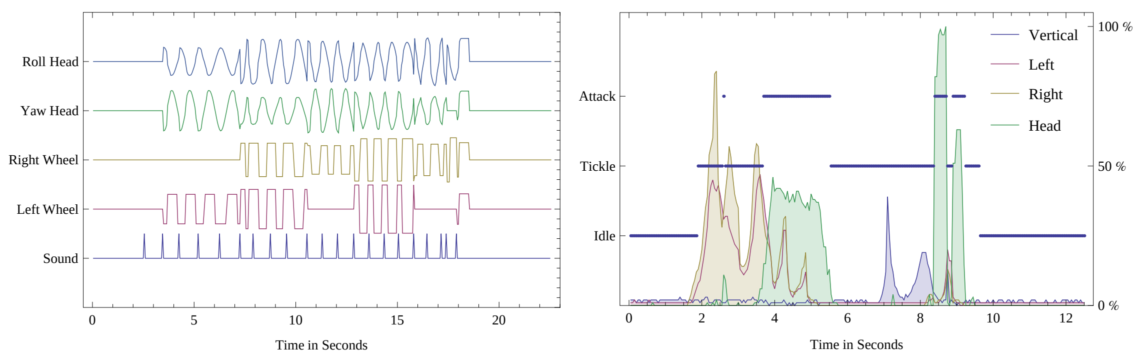
The objective of this example is to illustrate the process of training a typical behavior. Fig. 6 shows an example of training an escape behavior. To begin training the programmable brick is placed on the train slot of the *Playte*. The robot is then taught to drive away (using the gamepad) whenever a hand is triggering its vertical distance sensor. When the hand is removed the robot is taught to stand still. The programmable brick is then moved to the top row slots of the *Playte* and the behavior is tested (autonomous without gamepad). We observe the successfully trained behavior in this typical example.

Additionally we have conducted a systematic experimental trails, which are not included in this paper due to space

Behavior Name	Basic Emotion	Start Condition	Stop Condition	Action
Tickle Play	Joy	Nearby	Not Nearby	Baby laughing and rapid movements with all motors.
Dance	Joy	Sound	No Sound	Head moves between different random poses to the rhythm of the sounds (e.g claps).
Parrot Song	Joy/Trust	None	None	Plays a happy children's song about a parrot (no movement).
Monkey Song	Joy/Trust	None	None	Plays a happy children's song about a monkey (no movement).
Follow	Trust	Nearby	Not Nearby	Follows the object in front of the robot, while still keeping a safe distance.
Play Dead	Anticipation	Sound	Timeout	Turns around itself while slowing down until it dramatically dies.
Seek	Anticipation	None	None	Moves randomly around. Stops and barks when something is visible.
Hide	Fear	Not Nearby	Nearby	Moves quickly forward until something is nearby. Then it hides by ducking its head.
Escape	Surprise	Visible	Not Visible	Moves rapidly in random directions away from nearby objects.
Attack	Anger	Visible	Not Visible	Moves in short jerky motions before doing an attack move forward.

TABLE I

OVERVIEW OF PREPROGRAMMED BEHAVIORS.



(a) Dance (b) Subsumption

Fig. 5. Example of interaction with preprogrammed behavior bricks. (a) The "Dance" behavior with sound input and actuator output shown. (b) "Tickle Play" and "Attack" behaviors are placed on the *Playte* with the latter having higher priority. The interaction illustrates the function of the subsumption architecture.

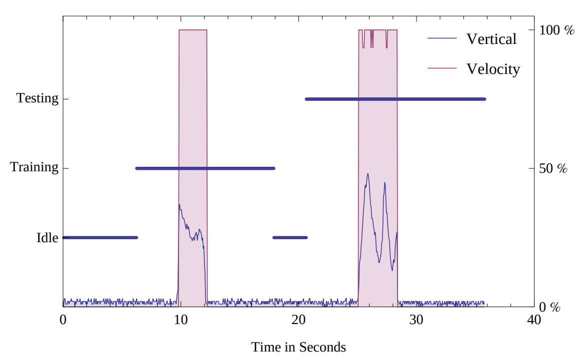


Fig. 6. Training example: The robot is trained to drive forward when a hand is activating its vertical sensor. Both the actual training and the following testing of the trained behavior is shown.

constraints. We can train the robot to do several different behaviors such as obstacle avoidance and following an object in front of the robot. In most cases the trained behavior will perform with a similar level of performance as while training. However, in around 30% of the trials the training fails even in fairly controlled experiments. We note that the main source of error is non-controlled interaction mistakes that were made while training the robot causing noise in the training examples. This high amount of mistakes is an indication of serious usability issues with the training system since even the trained experimenter would make such mistakes in fairly controlled setups. However, during the play session we did observe that the children were able to train the robot successfully.

C. Play Session

The objective of this play session is to improve our understanding of the advantages and disadvantages of the current design of the *Playte*. Which modes of interactions are more appropriate for children and how we can improve the design of the *Playte* system to increase its playfulness?

In this paper we report on two play sessions conducted with groups of two children, see picture in Fig. 1. The first group consisted of a boy and a girl, respectively 6- and 7-

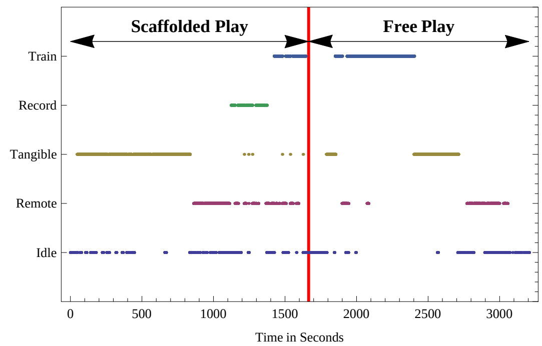


Fig. 7. Example timeline of interactions modes used during a play session with children aged 6 and 7 years-old

years-old. The second group also consisted of a boy and a girl both 11-years-old. The sessions were performed in quiet rooms at two different after school clubs in the Copenhagen area (Denmark).

We used a scaffolding play methodology to conduct the play sessions that builds on Wood et al. definition of scaffolding when learning [26]. The experimenter would guide the children through the different interaction modes in a structured fashion to teach them how to use the system. The children were free to play and experiment with the system throughout the process. The transition from one interaction mode to the next was dependent on our evaluation of the children's style of play. Once the children understood the interaction mode and their play activities were reducing focus the experimenter would progress to explain the next interaction mode. Halfway through the session the children were allowed to play freely for us to observe their understanding and motivations for using the different interactions modes.

A timeline of one play session is shown in Fig. 7. The data was logged digitally 20 times pr. second. The figure illustrates the sequence of interaction modes that were introduced to the children: i) tangible programming using the preprogrammed behavior bricks. ii) Direct remote control using a gamepad. iii) Recording and playback of new behavior bricks using the gamepad. iv) Training and combining new behaviors using the gamepad and sensory interactions.

1) Key Observations:: In both play sessions, regardless of age, the children seem to understand and were able to use tangible programming, record and playback, and training of the *Playte* system. However, only the older children seemed to have a good understanding of the effect of the subsumption architecture when combining preprogrammed behaviors and both groups of children clearly had difficulties correctly using the training and combining modes. From Figure 7 we see that no single modes were obviously favored. Except for the record and playback behavior in the sessions with the younger children, the children tried all the different ways of using the system again during the free play session. This seem to indicates that exploring the systems possibilities creates the most immediate enjoyment.

Both groups of children enjoyed playing with the system for the entire duration of the test. Their style of play were highly explorative, trying out the possibilities of the system. There was a clear difference in play approach between the younger and the older children.

The younger children were more playful and immediately started to treat the robot as a pet animal: talking to it in a soft voice and petting it with their hands. The younger children enjoyed making use of the subsumption architecture, even though they most likely did not understand it. They were very amused when the robot surprised them, and they realized that when placing the maximum number of preprogrammed behaviors on the *Playte* they could make a completely chaotic behavior that would do just that. Furthermore, the younger children used the robot as a social mediator, where one child would remote control the robot and interact through the robot with the other child.

The older children, on the other hand, approached the robot with a curiosity of understanding how the system worked and what they could make it do. They were more structured and systematic in their play approach and thereby progressed noticeably faster through the different means of interaction.

VII. Discussion

In this section we discuss the pros and cons regarding the different modes of interaction based on our experience developing and testing the *Playte*.

Direct remote control based on a gamepad is easy for the children to engage in. As it is direct it provides all the necessary feedback for them to quickly understand how it works. Furthermore, the gamepad interface is something that is well known to the target group.

Tangible programming with preprogrammed behaviors immediately triggers curiosity in children as they try to understand what the behavior does. This is well suited for quick fun, but the fun does not necessarily last long after the behavior is understood. The subsumption architecture prolongs the exploration process. For more complicated behaviors the explorative play may be longer as it is not as easy to deconstruct.

Programming by demonstration based on gamepad recording and playback works well for small customizations. As an example the older children learned the robot to do a specific dance using this technique. This functionality is well suited for reconfigurable systems, like the Topobo [17], where the focus is on the physical rebuilding while the record functionality is a quick way of making your creation come to life. However, the behaviors themselves quickly becomes monotonous as they are non-interactive and easily predictive.

Programming by training behaviors opens up for many possibilities that potentially can make it easy for children to program an exact behavior for the robot. However, in its current implementation, it is too difficult to use. It demands a solid understanding of how the sensors work and what the robot is able to do with its actuators. Furthermore, it introduces some usability issues as touched upon in the experimental evaluation. When training the robot for a specific sensor input, all other sensor values must be kept constant. This proved to be very difficult for a mobile robot. It is also advisable to be careful not to introduce the training method to children as a mean to learn the robot anything they want. They might disappointed when they cannot learn it to fly, play football or play rough-and-tumble.

VIII. Conclusions and Future Work

In this paper we described the design of the tangible interface *playte* with several modes of interaction, i.e. direct remote control, tangible programming, programming by demonstration, and programming by training. We experimentally exemplified the underlying behavior arbitration and training strategy. Further, we presented key observations from play sessions and observed that the *Playte* is appropriate and engaging for children aged 6 to 11. Finally, we discussed the pros and cons of the four interactions modes evaluated.

In future work we will port the *Playte* to our educational modular robotics system [16], [7] in order to analyze the different interactions modes on a modular robot that enables the children to create their own robots. Further, we will add visual programming language support to the *Playte* to enable a more complete comparison between different interactions modes by enabling the children to create their own behaviors with a more symbolic approach.

IX. Acknowledgments

This work was performed as part of the "Modular Playware Technology" project funded by the Danish Advanced Technology Foundation and as part of the "Assemble and Animate" project funded by the Danish Council for Independent Research (Technology and Production Sciences). Thanks to Mikael Moghadam and members of Center for Playware for their contributions to the project.

References

[1] A. N. Antle, M. Droumeva, and D. Ha. Hands on what?: comparing children's mouse-based and tangible-based interaction. In *8th International Conference on Interaction Design and Children (IDC09)*, pages 80–88, 2009.

[2] R. Brooks. A robust layered control system for a mobile robot. *Robotics and Automation, IEEE Journal of*, 2(1):14–23, 1986.

[3] D. J. Christensen, U. P. Schultz, and M. Moghadam. The assemble and animate control framework for modular reconfigurable robots. In *Proceedings of the IROS Workshop on Reconfigurable Modular Robotics: Challenges of Mechatronic and Bio-Chemo-Hybrid Systems*, 2011.

[4] C. Eickhoff, C. G. Harris, A. P. de Vries, and P. Srinivasan. Quality through flow and immersion: gamifying crowdsourced relevance assessments. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 871–880. ACM, 2012.

[5] R. Fogh and A. Johansen. The play grid. *Proceedings of the 12th International Conference on Interaction Design and Children - IDC '13*, pages 356–359, 2013.

[6] E. Hatfield and J. T. Cacioppo. *Emotional contagion*. Cambridge university press, 1994.

[7] B. Heesche, E. MacDonald, R. Fogh, M. Pacheco, and D. J. Christensen. Playful interaction with voice sensing modular robots. In *Proceedings of the International Conference on Social Robotics (ICSR2013)*, 2013.

[8] M. S. Horn, E. T. Solovey, R. J. Crouser, and R. J. K. Jacob. Comparing the use of tangible and graphical programming languages for informal science education. *Proceedings of the 27th international conference on Human factors in computing systems CHI 09*, 32:975, 2009.

[9] H. Ishii and B. Ullmer. Tangible bits: towards seamless interfaces between people, bits and atoms. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '97, pages 234–241, New York, NY, USA, 1997. ACM.

[10] C. Jessen and H. H. Lund. On play forces, play dynamics, and playware. Unpublished manuscript.

[11] D.-Y. Kwon, H.-S. Kim, J.-K. Shim, and W.-G. Lee. Algorithmic bricks: A tangible robot programming tool for elementary school students. *Education, IEEE Transactions on*, 55(4):474–479, 2012.

[12] H. H. Lund. Modular interactive tiles for rehabilitation: evidence and effect. In *Proceedings of the 10th WSEAS international conference on Applied computer science*, ACS'10, pages 520–525, Stevens Point, Wisconsin, USA, 2010. World Scientific and Engineering Academy and Society (WSEAS).

[13] H. H. Lund and J. D. Jessen. Effect of playful balancing training-a pilot randomized controlled trial. *2013 ISAROB*.

[14] T. S. McNerney. *Tangible Programming Bricks : An approach to making programming accessible to everyone*. PhD thesis, 2000.

[15] M. P. Michalowski, S. Sabanovic, and H. Kozima. A dancing robot for rhythmic social interaction. In *Human-Robot Interaction (HRI), 2007 2nd ACM/IEEE International Conference on*, pages 89–96. IEEE, 2007.

[16] M. Pacheco, M. Moghadam, A. Magnússon, B. Silverman, H. H. Lund, and D. J. Christensen. Fable: Design of a modular robotic playware platform. In *Proceedings of the IEEE Int. Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, May 2013.

[17] H. S. Raffle, A. Parkes, and H. Ishii. Topobo: a constructive assembly system with kinetic memory. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '04, pages 647–654, New York, NY, USA, 2004. ACM.

[18] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, and Y. Kafai. Scratch: programming for all. *Communications of the ACM*, 52(11):60–67, 2009.

[19] L. P. Rieber. Seriously considering play: Designing interactive learning environments based on the blending of microworlds, simulations, and games. *Educational technology research and development*, 44(2):43–58, 1996.

[20] N. Rusk, M. Resnick, R. Berg, and M. Pezalla-Granlund. New pathways into robotics: Strategies for broadening participation. *Journal of Science Education and Technology*, 17(1):59–69, 2008.

[21] E. Schweikardt and M. D. Gross. roblocks: a robotic construction kit for mathematics and science education. In *Proceedings of the 8th international conference on Multimodal interfaces*, ICMI '06, pages 72–75, New York, NY, USA, 2006. ACM.

[22] O. Shaer and E. Hornecker. Tangible user interfaces: Past, present, and future directions. *Found. Trends Hum.-Comput. Interact.*, 3:1–137, January 2010.

[23] K. Stoy, D. Brandt, and D. J. Christensen. *Self-Reconfigurable Robots: An Introduction*. Intelligent Robotics and Autonomous Agents series. The MIT Press, 2010.

[24] H. Suzuki and H. Kato. Interaction-level support for collaborative learning: Algoblock—an open programming language. In *The First International Conference on Computer Support for Collaborative Learning*, CSCL '95, pages 349–355, Hillsdale, NJ, USA, 1995. L. Erlbaum Associates Inc.

[25] L. S. Vygotsky. *Mind in society: The development of higher psychological processes*. Harvard university press, 1980.

[26] D. Wood, J. S. Bruner, and G. Ross. The role of tutoring in problem solving*. *Journal of child psychology and psychiatry*, 17(2):89–100, 1976.

[27] P. Wyeth and G. Wyeth. Electronic blocks: Tangible programming elements for preschoolers. *Proceedings of the Eighth IFIP TC13 Conference on Human-Computer Interaction*, pages 496–503, 2001.

[28] L. Xie, A. N. Antle, and N. Motamedi. Are tangibles more fun?: comparing children's enjoyment and engagement using physical, graphical and tangible user interfaces. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages 191–198, 2008.