Style by Demonstration: Teaching Interactive Movement Style to Robots

James E. Young[1,2], Kentaro Ishii[2,3], Takeo Igarashi[2,3], Ehud Sharlin[4]
young@cs.umanitoba.ca, kenta@ayu.ics.keio.ac.jp, takeo@acm.org, ehud@cpsc.ucalgary.ca

[1]University of Manitoba, Winnipeg, MB, Canada
[2]JST ERATO, Tokyo, Japan
[3]The University of Tokyo, Tokyo, Japan
[4]University of Calgary, Calgary, AB, Canada

ABSTRACT
The style in which a robot moves, expressed through its gait or locomotion, can convey effective messages to people. For example, a robot could move aggressively in reaction to a person's actions, or alternatively react using a set of careful, submissive movements. Designing, implementing and programming robotic interfaces that react to users' actions with properly styled movements can be a difficult, daunting, and time consuming technical task. On the other hand, most people can easily perform such stylistic tasks and movements, for example, through acting them out.

Following this observation, we propose to enable people to use their existing teaching skills to directly demonstrate to robots, via *in-situ acting*, a desired style of interaction. In this paper we present an initial style-by-demonstration (SBD) proof-of-concept of our approach, allowing people to teach a robot specific, interactive locomotion styles by providing a demonstration. We present a broomstick-robot interface for directly demonstrating locomotion style to a collocated robot, and a design critique evaluation by experienced programmers that compares our SBD approach to traditional programming methods.

Author Keywords
social human robot interaction, programming by demonstration, style by demonstration, tangible user interfaces, locomotion style, qualitative evaluation

ACM Classification Keywords
H.5.2 [Information Systems] Information Interfaces and Presentation: User Interfaces – Input devices and strategies

General Terms
Algorithms; Design.

INTRODUCTION
When designing robots that enter people's everyday environments it is important to look beyond utility and functionality and to explicitly consider *style*: the "expressive movement," way in which an action or behavior is performed [11]. People care a great deal about the style of objects and technologies that they possess, and a design can directly affect a person's interaction experience and satisfaction with things, including robots [26, 37]. We argue that people will desire their robots to be stylish, attractive and pleasing similarly to how they desire an attractive table, wristwatch, or car. This insight, we believe, adds an important style and design layer to the more-traditional utility and goal oriented human-robot interaction (HRI) approach. In this paper, we focus on the style of basic locomotion-based robotic interaction, but we believe that our approach can scale beyond this fundamental layer.

Figure 1. Style by demonstration: (left) a person uses a broomstick interface to show the robot a specific style of interactive movement (in this example, *follow politely*), and (right) the robot mimics the demonstration to follow politely.

Previous research shows that simple movement patterns by purely abstract entities lead people to construct intricate stories, personalities, and emotions [17]. Thus we point out the importance of considering the style of a robot's interactive locomotion movement as a key interaction layer, arguably as crucial to HRI design as a robot's physical form. We view interactive locomotion *style* as the manner in which a robot moves around a space in real-time reaction to a person. This stylistic locomotion can provide the robot with a body-language vocabulary, allowing it to interact with people in stylistically-appropriate ways in different tasks and settings. For example, a robot can present itself as being confident when approaching a task, or perhaps confused and unsure of its understanding, via upbeat or hesitant movements respectively. A robot which may endanger people can warn them to keep away by moving in aggressive, jerky movements, or a servant robot can inform people of its aspiration to serve by politely and humbly walking slowly behind them.

IUI'12, February 14—17, 2011, Lisbon, Portugal.

Advanced programming is generally required to enable robots to perform even simple tasks in the real world (e.g., pick up objects or follow a person), in part because robot actions must interactively respond to dynamic and unpredictable environments and people in real time. In many such real-world HRI tasks the style of the resulting robotic actions tends to reflect programming intrinsic task-completion- and efficiency-oriented approach [36]. To create robots that not only perform their tasks, but also act with specific style, for example, an aggressive, timid, or careful fashion, further complicates the HRI design and implementation challenge.

We propose a novel social human-robot interaction technique we call style-by-demonstration (SBD) as a means to enable people to directly design the style of interactive robot movement by providing an exemplar. This ueses in-situ physical acting to bypass traditional programming. Teaching in general is a skill that people are intimately familiar with from everyday interaction with other people, part of humanity's social stock of knowledge [2]. Teaching movement style via acting can be an intuitive and powerful method for showing a robot the style of how to perform a task. Our approach takes advantage of these existing teaching skills in order to empower people and to alleviate the difficult task of programming the style of interactive robotic movement behaviors. Programming-by-demonstration was previously explored in robotics (e.g., [10]), but as far as we know only targeted the fulfillment of a goal-oriented task. Our proposed SBD is unique in its focus on teaching the *style* of the interactive robotic behavior, rather than a set of actions leading towards an explicit task goal.

In this paper we present the design and implementation of an SBD system which enables people to teach a robot how it should interact with a person, specifically, the interactive style of the robot's locomotion. Our prototype is based on a robot-on-a-broomstick tangible interface (Figure 1) and we realized the learning by adapting and expanding an existing animation-only SBD system (*Puppet Master*, only for animated entities [38]). To evaluate our system our paper includes a design critique performed by experienced programmers who compared our SBD systems with traditional programming techniques. Overall, we believe that our SBD technique offers a new insight on how social human-robot interaction can be designed using in-situ style demonstration and acting. We believe that systems similar to ours will be increasingly important as robotics and human-robot interaction (HRI) advances, and as more people with no programming background integrate robots into their social work and domestic environments.

RELATED WORK

There has been extensive work that shows how people attribute simple geometric shapes such as triangles, circles, and points with intentionality, personalities and gender based solely on the style of their movements around a screen [8, 17, 21, 29, 34]: for example, as an aggressive male *triangle* trying to corner a female *circle* [17]. Further, much of Disney's success at creating believable characters has been attributed to their realization that a character's movement style is critically important [32]. This has been an area of interest, for example as with one animated entity which performs scripted actions such as "pick up a glass" or "knock on a door" in a "neutral," "shy," or "angry" fashion [1]. This paper builds on these extensive past efforts, and attempts to adapt them to the field of HRI.

Style work perhaps most similar to ours is that which considers robot behavior in proximity to people. Although this primarily targets practical engineering, mobility and vision challenges [5, 23] or goal-centric predictive models [6], others looked at how close a robot should be to a person for comfort [35] and how a robot should follow naturally [12] (i.e., copying a path versus shortest route). Our work continues this research direction by developing a technique that allows users to easily teach robots how to interact in a desired style.

The most direct method employed for the creation of interactive robotic behavior is to explicitly program the behavior model, defining how the robot should react in particular situations (e.g., as with [4]). This is also true for the related problem of programming interactive animated characters such as those found in video games [3, 20] – such autonomous characters must also solve real-time interactive behavior problems, and so techniques from animation are highly relevant for robotics. While the programming approach results in satisfactory behaviors, it is generally inaccessible to the non-technical end-users who will want to customize their robots' behaviors.

Programming by demonstration is a mature field of research that has been applied to such applications as programming GUI operations [25] and animation, for example, for interactive collision avoidance and planning [7], or for the direct control or static playback of animations [18]. Many such approaches require synthesis from extensive example databases that require large amounts of technical-user pre-processing [8, 22]. For robots, similar approaches are used to teach navigation routes [20], moving or posing in a human-like fashion by copying interactive behavior (not learning it) [23], or performing specific physical tasks [13, 16, 24, 27]. Overall, these approaches target a particular task goal, movement, or plausibility of physical motion such as realistic walking or collision avoidance. Our work is unique in that it does not attempt to teach the steps needed for achieving a specific goal, nor how to copy a specific interaction flow, but rather concentrates on teaching the high-level *style* of interactive behavior, recognizing interactive style as its main focus in a manner detached from a specific task goal.

Some programming-by-demonstration robot systems [10, 28] enable the creation of stylistic and expressive motions, but these result in static movements only and are not interactive to a person's input. Other work incorporates prescripted style and emotion into their learning interfaces [24], but these are statically used to represent algorithmic states,

and are not learned from demonstration. Yet other work focuses on understanding how people teach each other to inform how to design robot-teaching interfaces [33]. However, as far as we know there is no previous work on explicitly teaching a robot interactive style.

Overall, while programming by demonstration is widespread in robotics and the importance of robot movement style is evident, our work is as far as we know the first to explicitly focus on the creation of *interactive* and *stylistic* robotic behaviors (i.e., *stylistic locomotion*) via demonstration, and the first to directly explore how such style can be interactively embedded within a robot's locomotion path.

STYLE-BY-DEMONSTRATION INTERACTION

Here we introduce our interaction design for robotic locomotion style-by-demonstration (SBD). Our SBD system has two phases, *demonstration* and *generation*. During *demonstration*, the user provides an example of how the robot should be interacting with the person in a desired style (a paired motion). Both paths are given simultaneously in real time: the user is demonstrating the robotic movement style in-situ, reacting to an actual person's movement, providing the robot with an exemplar of what it should react to, and in what style it should react (Figure 1).

After *demonstration*, in the *generation* phase the robot incorporates the learned reactionary characteristics into real-time interaction: the person moves freely as before but now the robot is autonomously interacting and reacting to the person's movement with its newly acquired and appropriated styled movement style (Figure 1). In our system, the robot's movement model is entirely contained in this learning: no prior robot movements or path following algorithms are included.

A Broomstick Interface for Teaching Movement Style

One leading goal in developing our new interface was to enable the user to focus on desired motion style rather than on the mechanics of moving the robot. In this case, the robot we use is an iRobot Create (Figure 2), a disc-like robot that moves via a 2-wheel-plus-caster base (preventing it from moving sideways) and rides close to the ground. We ruled out directly tracking a person's natural movements for demonstration as they would likely be too expressive and contain motions and nuances not reproducible by the robot. Further, the small size and height of our robot makes it impractical for a person to grab and move directly (i.e., will likely require crawling). Rather we suggest using a replica of the robot as an interaction metaphor supporting the in-situ demonstration of movement style.

Figure 2. iRobot Roomba, with underside view: two-wheel base and coaster highlighted.

To implement our approach we developed a broomstick tangible-user interface (TUI) that connects directly to an iRobot Create, allowing the user a direct yet valid method of demonstrating style, encapsulating the robot's movement properties and constraints [14, 15]. This further focuses on the robot's particular embodiment, enabling the person to experience the interaction through the robot's capabilities, and forcing them to adhere to them [9]. Our design ensures the user's demonstrated motions will be later reproducible by the robot, and at the same time communicates the mechanical movement constraints to the user as they engage in the movement demonstration. This immediate tactile feedback reduces indirection between the input device and ultimate output, improving usability [19, 30].

We use a regular aluminum broomstick attached to the robot (Figure 1, Figure 3, Figure 9) via a two-axis swivel, allowing the stick to be freely moved (but not twisted) without moving the robot. A user can turn the robot by tilting the broomstick left or right, or twisting it, during movement. The robot's wheel motors have been disconnected (gears removed) to reduce friction and force required to push the robot. The result is a natural and familiar mechanism allowing the user to demonstrate interactive locomotion style to the robot.

Figure 3. Broomstick mounting.

During the demonstration phase, one person walks naturally in the space (for the robot to react to) while the user maneuvers the broomstick interface to demonstrate a following movement style to the robot (Figure 1, left side). When demonstration is finished, a completely autonomous (non-broomsticked) robot enters the space and follows the person using the movement style that was demonstrated (Figure 1, right side). Communication with both the broomsticked and non-broomsticked robots is handled via Bluetooth, and both robots' and person's locations are tracked using Vicon camera motion-tracking (Figure 4). IR-reflective markers were affixed to the robots and person's shoes, with the person's location approximated as the midpoint between shoes. Note that the user manipulating the broomsticked robot is not being tracked directly, but rather their demonstrated movement style is tracked continuously through the broomsticked robot.

ALGORITHM

We adapted and extended an existing animation-only algorithm called Puppet Master for demonstrating stylistic mo-

tion to animated characters [38]. Puppet Master is an SBD platform for creating paired, style-oriented behaviors; instead of being used for robots, however, the original Puppet Master targets unconstrained animated characters.

Figure 4. The interaction space, tracking cameras highlighted.

Puppet Master is based on the notion of stylistic and interactive locomotion paths: the way the character moves to interact with a counterpart character [38]. As explained above for robotic SBD, Puppet Master has both *demonstration* and *generation* modes. After *demonstration* the real-time relationships between two animated characters are examined via their relative positions, orientations and velocities, all over an interaction history. For *generation* Puppet Master then uses a pattern patching approach for real-time output of the autonomous animated interactive character to employ the reactive style as demonstrated [38].

Puppet Master is a pattern matching algorithm that runs at relatively high speed (40hz [38]) to achieve interactivity, and uses heavy smoothing and frequency-analysis filtering to maintain coherent results. First, for *demonstration*, users provide an example of one animated character interacting with another using the locomotion path only, e.g., one character stalking another; this was previously accomplished using a digital tabletop system with tangible pucks for simultaneous character manipulation [38]. Then, during *generation*, Puppet Master searches the training set for locomotion paths similar to the real-time state, to inform ongoing character movements. Thus, the animated character output is a patch-work of training data pieces (Figure 5). Training-data is searched every time step to incorporate ongoing real-time user input, and uses a window of data (1s) compared against a window moving over the training

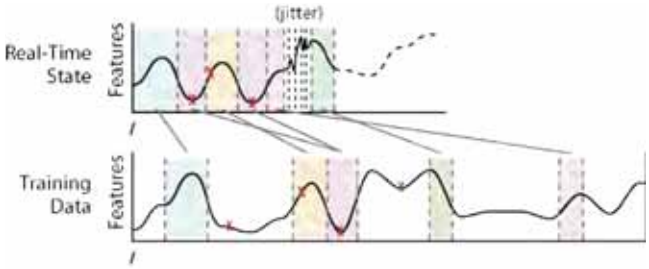


Figure 5. Real time entity is a set of patches from the best-match training data (illustration only). All dimensions of training and output for both the leader (human, in our work) and the reactor (robot) characters are summarized into a single curve for clarity. The red "x"s denote discrete auxiliary action events.

data set: this history is important to maintain higher-level movement coherency, e.g., one entity circling the other.

Here we explain the comparison metric. First, locomotion paths are distilled into defining scalar features: velocity, turning amount (left or right), relative look direction (one looking at the other), and relative position (e.g., behind and to the left of the other). These features form a vector at each time point that can be compared. The best-match training data is selected by minimizing the Euclidean distance between the real-time window of data and the moving window over the training data.

The best-match training cannot be copied verbatim to character output as the features generally conflict, e.g., the required movements to match target velocity (from the best match) may contradict the relative position. Puppet Master attempts to compromise by balancing the demands from the various features.

A problem with the Puppet Master algorithm is movement jitter, where the character will appear to vibrate rapidly in a way not consistent with the training data. This happens when the best-match source training data changes rapidly between conflicting source locations, when their match scores are similar; thus, rather than using coherent patches from the training data, in these instances only very small slices are used: this phenomenon is denoted in Figure 5. Puppet Master's solution was to a) heavily smooth the robot motions to filter the jitter and b) use frequency analysis to re-introduce the appropriate training movement detail that was lost in the filtering. This provided an improvement but the results were still not satisfactory to users [38].

The summary provided here was for the purpose of explaining the background to the Puppet Master platform; important details were omitted for brevity and we recommend those wishing to implement this work to refer to the original Puppet Master papers.

Adaption to Robots

When we applied the animation-only Puppet Master algorithm to robots we quickly found that it was not successful in generating reasonable stylistic robotic motion due to the hard movement and speed constraints posed by real robots. The animation Puppet Master freely modified the output movement (i.e., smoothed and exaggerated or tweaked) where necessary to best match the desired overall result; for example, an animated character could easily be moved a little sideways or turned while moving toward an appropriate relative location or orientation. Robots, however, work on irregular surfaces and must adhere to real constraints such as movement speed or physical design. Robots cannot be moved as freely as the animated characters, and we found that in our case the robot could not move as required to match the Puppet Master algorithm output; problems include the robot's acceleration and deceleration delays, and in our case the platform's inability to move sideways.

Thus, the result of our initial algorithm adaption was that the robot lost both the movement detail (or texture) and the

relative-to-human position prescribed by the Puppet Master algorithm. Overall, the resulting reactive movement style was unconvincing and our attempts to reduce the algorithm from the animation's 40 Hz [38] to 20 Hz to better reflect the robot limitations had no observable improvement.

Our solution was to thoroughly revise the adopted Puppet Master algorithm. We applied rudimentary frequency analysis grounded in the robot's movement capabilities, balancing overall robot-human localization (low-frequency) and movement texture (high-frequency component). We used a kinematic model of the iRobot Create – describing the robot's movement capabilities and constraints (Figure 6) – to produce both the high and low frequency components in terms of robot commands: the results are movements guaranteed to be reproducible by our robot. We used this model (Figure 6) by solving for robotic movement commands (*velocity* and *turning radius*) given a target $\Delta x, \Delta y,$ and $\Delta\theta$ (difference in look direction) using simple trigonometry: $r = \Delta x + \frac{\Delta y}{\tan\theta}$, $vt = \theta r$ where r is turning radius and vt is distance or velocity multiplied by time.

First, we use the kinematic model to solve for the robot command that will move it to a given Puppet Master target state (Figure 7a,b). This is the *low-frequency* component of the desired movement as it changes relatively slowly over time, which places the robot in the desired area and orientation. However, as the robot cannot instantly keep up with the moving target over time, the texture (or details) of the desired movement are lost. Second, we find how the robot can reproduce the exact desired path (Figure 7c), by solving for the delta movement (change in direction, and location) rather than the target location. This maintains the high-frequency component, but drift in the robot's imperfect movements means that the robot soon loses its relative localization: the *low frequency* component.

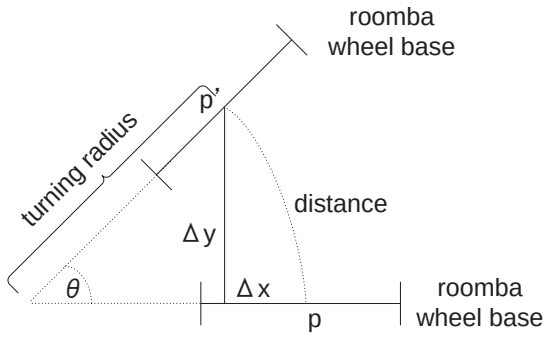


Figure 6. The Roomba movement capabilities, showing the relationship between its control scheme of setting *velocity* (distance / time) and *turning radius* and the resulting Δ x and Δ y.

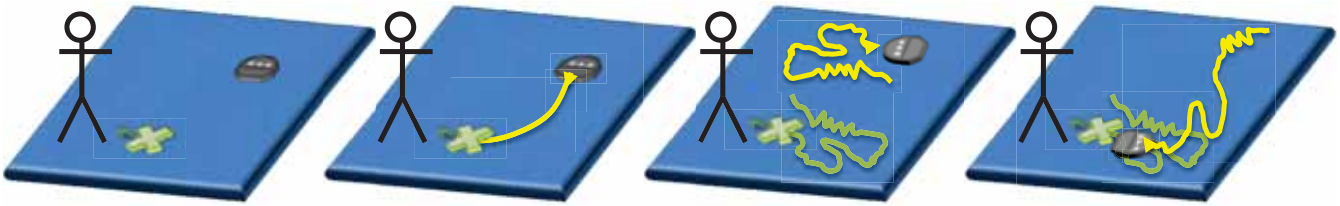
Finally, we combine the above two components by using a weighted average of the two resulting robot commands, the velocity and turning radius Figure 7d. Focusing on the high-frequency component results in better texture retention but more location drift, and a focus on the low-frequency component has the opposite effect. We use a 70/30 high/low-frequency balance, selected through experimentation. The intuition is that existing high-frequency robot movements which tend to move away from the target location are slightly modified by the algorithm to correct their direction, while movements that are already directed toward the overall target are generally unchanged. Figure 8 details the overall process. During *demonstration*, input is fed directly into the Puppet Master base algorithm. For *generation*, the Puppet Master output is filtered through the kinematic model as a form of frequency analysis, and both outputs are weighted to produce the final robot command.

We believe that our solution is a unique method for filtering a desired movement against robot capabilities while focusing on the balance of texture and locomotion, and can be used for the general problem of applying unconstrained animation behaviors to the rigid constraints of real robots.

We further propose that our algorithm could be applied to different robot morphologies simply by using the proper kinematic model. This is not necessarily for locomotion paths only, but for any movement path, e.g., an arm path or head movement path. If a robot system is given an unpredictable movement path without look-ahead capabilities, it can apply our method to adopt the path while focusing on maintaining style if it has a kinematic model which enables it to: a) calculate the closest possible fit to reproducing that path (via a kinematic model) and b) calculate the shortest-path route to a far absolute target. These can be combined as explained above for real-time dynamic robot motion. Thus, we believe that our system can be extended and used to demonstrate movement style to robots that are more complex, with more degrees-of-freedom than the platform currently used.

Integrating Stylized Auxiliary Actions

In addition to the above algorithm enabling the demonstration of stylized movement to robots, we explored whether



(a) robot and target state

(b) solve for robot to reach target state in one time step (if possible)

(c) target state over time forms path, solve for robot to reproduce texture

(d) combine movements, tend toward target location over time

Figure 7. The algorithm robot-movement frequency analysis principle.

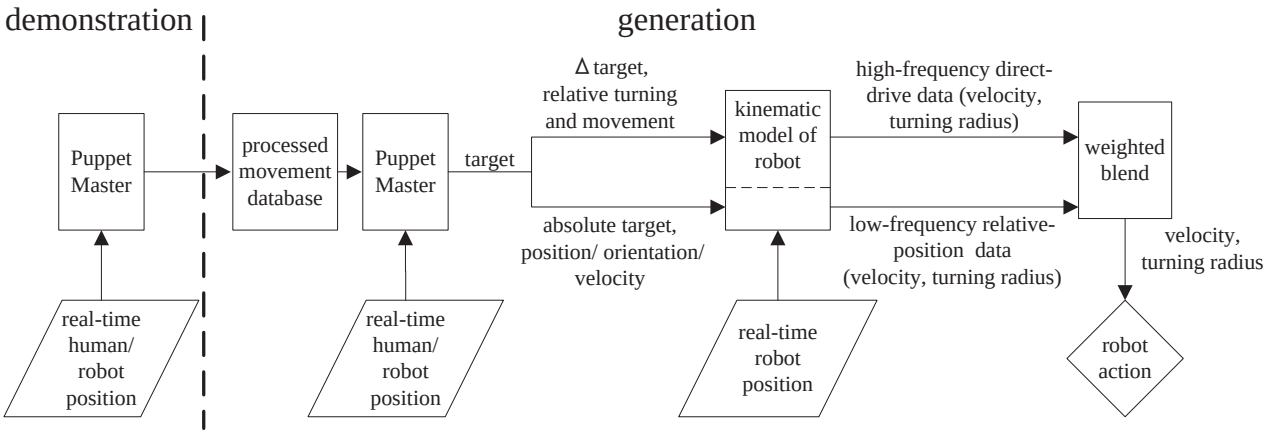


Figure 8. The algorithm data-flow and processing, and how Puppet Master connects to the real-time robot data.

the algorithm can be extended beyond movement, to support stylizing any discrete robot actions. Our presented extension allows the user to teach the robot the *appropriate times* during interaction in which it should make *happy* or *unhappy* sounds in relation to the overall behavior style. To support the teaching of auxiliary actions by the user, we augmented the broomstick with two buttons (Figure 9) so that the person can trigger the robot sounds (happy or sad) during demonstration. The buttons are wired into a modified Phidget Remote wireless FOB clicker and standard Phidget interface kit.

Figure 9. Broomstick-Roomba interface, with mounted buttons.

The main purpose of this extension goes beyond sound, but rather aims at exploring the scalability of the SBD algorithm and interface to any robotic actions which are not necessarily derivative of locomotion paths. Algorithmically and interaction-wise, the current robot sounds can be replaced by any discrete pre-programmed robotic action such as taking a photo, facial expressions, discrete gestures, or speech. Following, our SBD approach could be used to teach a robot the difficult-to-define problem of *when* to perform such discrete actions.

Algorithmically, we accomplished auxiliary actions by associating the demonstrated actions with the existing training data on the time axis, such that the data is marked with the action at the appropriate time: a visualization of how the timings are superimposed on the training data is given in Figure 5. Then, as particular training data is used to construct output during generation, the associated actions are included in the robot output.

EXPERIENCED-PROGRAMMERS DESIGN CRITIQUE

While the fundamental premise of our work is that SBD provides a new interaction mechanism that allows untrained users to easily design stylized robotic movement, we focus this design critique on a more preliminary question: how does SBD differ from more-traditional programming methods? We recruited experienced programmers and asked them to create interactive robot behaviors both programmatically and using our SBD broomstick interface. These participants were qualified enough to create interactive behaviors using programming, and can engage the SBD interface the same as any other person, giving them a unique comparison vantage point and providing us with an analysis of the trade-offs and benefits of each. This further enables us to reflect on a more acute question: if an expert were to design a robotic interactive style, will they still find advantages in SBD compared to the more structured programming methodologies?

We expect that these demanding and insightful technical users would inform us on a group of users that can be viewed as a worse-case subset: those with high and critical expectations of what the robot should do. Arguably, if experienced programmers who can design robotic behaviors via programming find our SBD approach advantageous, regular users lacking such expertise and training will also be able to benefit from the SBD method.

For our design critique we recruited four experienced programmers from our graduate-student lab. Our participants were three male / one female and did not have prior exposure to our work. The programmers were asked to create stylized robot movement behaviors via two conditions: 1. programming using Java code, and 2. creating the same stylized movement behaviors using our broomstick SBD tangible interface, and to reflect on their experiences with both. This design critique was primarily exploratory, and thus our main analysis method was to provide initial qualitative detailed descriptions of the reflections as themes to portray the experiences and opinions of the participants (following an evaluation method described in [31]): we see this as a precursor to direct more formal experiments.

Study Design and Procedure

We selected four stylized robot movement behaviors, a polite follow (*polite*), a robot stalking a person (*stalker*), a

robot that is happy to see the person (*happy*), and a robot that is attacking a burglar (*burglar*).

For the programming condition we provided a robot-simulation prototyping test-bench and an API for reading the robot and person's locations and providing real-time commands; Figure 10a shows the output screen with robot, person (happy face), and an arrow handle for manual control. After initial explanation, programmers were given two hours to create the four behaviors.

Following the programming condition, participants created the same stylized movement behaviors using the broomstick SBD interface, where they were allowed to demonstrate stylized movement behavior, and observe it repeatedly for as long as they wished before moving on to the next, and could re-train a behavior if they were not satisfied with the result (Figure 10b). This phase was video-taped.

Post-study we interviewed (unstructured) the participants regarding their overall experience.

Results

All participants took the full two hours to program their stylized movement behaviors, and all were able to create their behaviors in the time given, although one programmer stated that they would require a great deal more time to implement proper "nuanced behaviors." For SBD, all programmers were observed to "act the characters," making faces, laughing, etc. Table 1 shows SBD demonstration attempts and length (seconds). The "total time" listed is the duration of creating all behaviors using SBD, from start to finish, including observation, thinking, brief discussion, and retraining time.

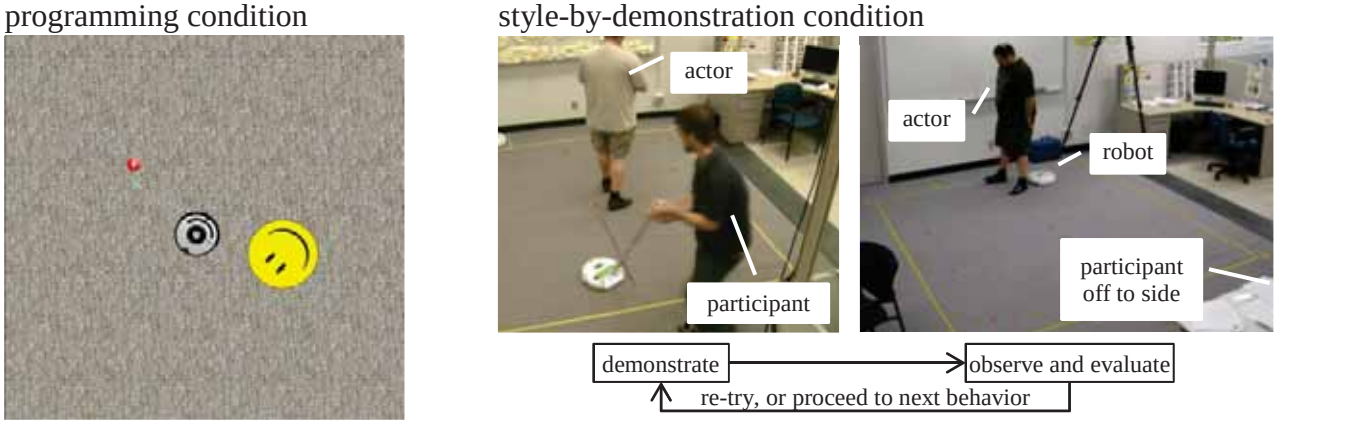
When asked which they preferred, direct programming or programming by demonstration, all programmers articulated a set of trade-offs rather than preference, for example, "the programming spoke to the scientist in me, and the other, the broomstick demonstration, spoke to the non-scientific part of me."

The programming approach was touted as being more accurate and kept the programmer "in control" in comparison with the demonstration. Because of this, one person stated that they felt like they had "a lot more power to do something creative." However, control is not easy or complete; one programmer noted "when you're programming something you have to anticipate ... what kind of situations can come up and how [the robot] should react ... that's not a natural way of doing things." The programmers made statements highlighting the difficulty of the direct programming condition. For example, general programming difficulties still exist: it's "hard to debug the program even though I have the simulated environment," "even when I program I don't know exactly what is going to happen," and "when I see problems, I still don't know why it happens."

Table 1. Demonstration times in the SBD condition; only three are shown (of four) as one participant requested not to be video-taped, and we did not record the data directly

Behavior	programmer		
	1	2	3
Polite follow – tries	2	1	1
time	44 s	31 s	24 s
Stalker – tries	1	1	1
time	65 s	46 s	31 s
Attacking burglar – tries	1	1	1
time	51 s	44 s	25 s
Happy to see you – tries	2	1	1
time	40 s	37 s	24 s
total time	14 m 49 s	8 m 52 s	7 m 40 s

Programmers mentioned that by focusing on style during programming the "types of things [they were] trying to express were more nuanced, more complicated behaviors" than the "easily expressed things like sine waves" that they are used to creating in interactive characters (referring to the approach to smooth animation). One programmer noted that doing the programming before demonstration helped to highlight the sheer difficulty of the real-time problem, and helped them to appreciate the demonstration system.



a) Programming test-bench.

b) Demonstrate and observe cycle for broomstick SBD.

Figure 10. Participants were given two hours with the test bench to program the *polite*, *stalker*, *happy*, and *burglar* behaviors. Following, participants used the broomstick SBD interface to create the same behaviors, one at a time.

Programmers noted that the SBD broomstick interaction is much faster and easier than direct coding. By using the demonstration system they "did not have to think technically or analytically," and could more-easily program movement styles. As such "there is a huge time-saving potential here." One reason cited for the broomstick's success is that people are very skilled at "understanding changing situations on an instant-to-instant basis and [can] essentially make up [their] own behaviors on the fly." However, several programmers pointed out that SBD cannot be perfect as they are "at the mercy of the system" and their "demonstration is just a small part of the bigger thing." They are "relying on its interpretations of [their] intentions, rather than on [their] actual intentions. There is no way to directly convey intentions," for example, they could not specify hard constraints such as "stay away from the corner."

We asked the programmers to provide an informal design critique of our broomstick interface. One programmer mentioned that the robot can be difficult to turn quickly; however, this programmer tried to train the robot to turn much-more quickly than the real robot could perform. That is, although the interface is limiting, its restrictions are in accordance with the movement capabilities of the real robot. Another, who has professional game development experience, said that he thinks of the robots as non-player-characters in a game where he would not consider using SBD as-is for entities at the forefront of the. However he felt that there is real potential for SBD for side-line entities. Another participant pointed out that the inherent inaccuracy of the demonstration system is not necessarily a problem, as perhaps the broomstick can be used to capture a rough behavior and serve as a medium fidelity prototyping method for behaviors that can be later programmed more thoroughly. Programmers also provided suggestions on how to mix the pure demonstration approach with more logical components, for example, to enable demonstrators to explicitly specify which components are important, or give them easy-to-understand variables to tweak when observing the result (e.g., as tangible knobs or a hand-held interface).

Two programmers noted that the physical energy required to demonstrate stylized movement using the SBD broomstick (in the large space) was more exhausting than the act of programming. They mused about the use of a remote controller to reduce the fatigue problem, although they both admitted that a remote controller would result in an interaction that would likely be less immersive and affording less expressiveness in comparison to the TUI broomstick. One participant suggested a tabletop TUI system as way to keep direct movement while lowering the effort required.

Discussion

The primary purpose of our design critique was to leverage the expertise of programmers and their unique perspective to analyze our SBD approach, particular interface and implementation. We believe that the detailed feedback from the programmers provides unique insight that will be valuable for the continued development and evaluation of SBD robotic interfaces.

Our design critique supports the general SBD idea and approach, suggesting that it is applicable and valid even for experienced programmers. As shown in Table 1, all participants managed to complete the creation and evaluation process in less than 15 minutes, substantially less than the 2 hours taken for programming. Obviously, we expect the SBD benefits to be dramatically stronger for users without extensive programming experience or abilities, where the SBD approach may be the only method that would enable them to design stylized robot movement behaviors

Beyond the time-efficiency results, we highlight the level of emotional engagement which emerged: the programmers using SBD readily acted and got into the characters, using their entire body to express movements, for example, laughing and making facial expressions to match what they were demonstrating. Thus, even for technical users who have an understanding of the nature of the robot, our results help support the idea that programming style to robots via demonstration leverages their innate social understanding and skills, and they employ emotional interaction similarly to how they would with a person.

One of the benefits of our evaluation study was that the participants, like any other person, benefited from the social stock of knowledge [2] that makes the SBD demonstration familiar and comfortable, and further had the benefit of technical understanding of the problem and the required expertise to solve it using programming. Their feedback improved our understanding of the accuracy / control versus time / ease trade-off. We observe that programming may enable people to be creative in ways that demonstrating does not, and that the complexity of the programming approach means that it still leaves a layer of uncertainty and mystery for experienced programmers, despite the extra control, which the SBD approach may help alleviate.

Further, our participants proposed to combine the programming (more control) and demonstration (easier to do) approaches, for example, by using the demonstration as a prototyping tool, or by including easy-to-understand parameters or conditions modifiable by the demonstrator.

Many of the participants' observations help us to better-understand the limitations of demonstration, for example, that there is potentially no optimal solution as machines cannot understand a person's intentions, only their actions, and this interpretation is subjective to the demonstration-learning interface and algorithm. We point out, however, that people suffer from the same problem as these robots: people cannot know others' intentions, only what can be deduced from interaction. Regardless, this suggests that we should aim to better understand the particular biases introduced by any given algorithm, and how this relates to target applications and usage scenarios.

Overall, this critique helped to verify the applicability of SBD for interactive robot behaviors and our particular in-

terfaces, and provided insight into the trade-offs between SBD and traditional programming.

CONCLUSION AND FUTURE WORK

In this paper we presented the idea of teaching robots *interactive, stylistic locomotion* by *demonstration*. We detailed our technical solution, introduced a novel robot-broomstick interface, and detailed our robotic evolution of an animation-based algorithm. Finally, we presented a design critique that helped us gain insight into the benefits and limitations of SBD in relation to traditional programming.

One deficiency of this research to date is the lack of a formal extensive study on the use of SBD for robots, the efficiency of the broomstick TUI, as well as the quality of the underlying robot SBD algorithm. The design critique presented in this paper serves as an important, but still preliminary step toward such a thorough evaluation.

Another issue for consideration is the concern that perhaps people may not be skilled at acting out how they want the robot to behave. That is, while people know what they like or what they want, they may not be able to demonstrate it in a manner which portrays their intent. Puppeteers and actors take years of training to develop these abilities.

In the longer term we are planning to explore the scalability of both our SBD approach and our current algorithm to more elaborate robotic expressions. For example, can we use a similar algorithm to simply demonstrate to a robotic teacher surrogate how to move its arms sternly, or enthusiastically? Can it be used to demonstrate to robotic service provider, say a waiter, how to collect dishes apologetically? We expect that designing style by demonstration can be mapped to other, more complex applications, and integrated closely with other robot programming approaches.

SBD provides a new language, a new vocabulary, which people can use to communicate and program robots: with existing methods, users who cannot program will be ignorant of how to teach their robot the desired style.

With the continued advance of technology robots are expected to share more and more of our physical and social spaces. Robot acceptance will be based as much on a robot's style of interaction as on their goal oriented task performance. Providing simple, well situated tools that will enable non-technical people to show their robots in what style to act can become an important layer of future efforts of integrating robotic interfaces in our everyday environments. We believe our style-by-demonstration approach presented here is an important early step in this direction.

REFERENCES

1. AMAYA, K., BRUDERLIN, A., AND CALVERT, T. Emotion from motion. In *Proceedings of Graphics Interface 1996. GI '96, Toronto, Canada, May 22–24, 1996* (1996), Canadian Human-Computer Communications Society, pp. 222–229.
2. BERGER, P. L., AND LUCKMANN, T. The social construction of reality: a treatsie in the sociology of knowledge. Anchor Books, 1966.
3. BLUMBERG, B. M., AND GALYEAN, T. A. Multi-level direction of autonomous creatures for real-time virtual environments. In *Proc. SIGGRAPH '95* (1995), ACM, pp. 47–54.
4. BREAZEAL, C., AND SCASSELLATI, B. A context-dependent attention system for a social robot. In *Proc. IJCAI '99* (Stockholm, Sweden, 1999), IJCAI Press, pp. 1146–1151.
5. BYERS, D., AND JENKINS, O. C. HRI caught on film 2: Hands-free human-robot interaction. In *Proc. HRI '08* (NY, USA, 2008), C. Bartneck, Ed., ACM.
6. CHUEH, M., JOSHI, S., YEUNG, Y. L. A., AND LEI, K. P. Towards a mobile-robot following controller using behavioral cues. In *Proc. ICIT '06* (USA, 2006), IEEE, pp. 150–157.
7. DINERSTEIN, J., EGBERT, P. K., AND VENTURA, D. Learning policies for embodied virtual agents through demonstration. In *Proc. IJCAI '07* (2007), IJCAI Press, pp. 1257–1262.
8. DONTCHEVA, M., YNGVE, G., AND POPOVIC, Z. Layered acting for character animation. *ÁCM Trans. Graph. 22*, 3 (2003), 409–416.
9. DOURISH, P. Where the Action Is: The Foundation of Embodied Interaction. MIT Press, MA, 2001.
10. FREI, P., SU, V., MIKHAK, B., AND ISHII, H. curlybot: designing a new class of computational toys. In *Proc. CHI '00* (NY, USA, 2000), ACM, pp. 129–136.
11. GALLAHER, P. E. Individual differences in nonverbal behavior: Dimensions of style. *Journal of Personality and Social Psychology 63*, 1 (1992), 133–145.
12. GOCKLEY, R., FORLIZZI, J., AND SIMMONS, R. Natural person-following behavior for social robots. In *Proc. HRI '07* (NY, USA, 2007), ACM, pp. 17–24.
13. GRIBOVSKAYA, E., AND BILLARD, A. D. Combining dynamical systems control and programming by demonstration for teaching discrete bimanual coordination tasks to a humanoid robot. In *Proc. HRI '08* (NY, 2008), ACM, pp. 33–40.
14. GUO, C., AND SHARLIN, E. Exploring the use of tangible user interfaces for human-robot interaction: a comparitive study. In *Proc. CHI '08* (2008), ACM, pp. 121–130.
15. GUO, C., YOUNG, J. E., AND SHARLIN, E. Touch and toys: new techniques for interaction with a remote group of robots. In *Proc. CHI '09* (2009), pp. 491–500.
16. HALBERT, D. *Programming by Example*. PhD thesis, University of California Berkeley, 1984.

17. HEIDER, F., AND SIMMEL, M. An experimental study of apparent behavior. *American J. of Psychology 57* (1944), 243–259.

18. IGARASHI, T., MOSCOVICH, T., AND HUGHES, J. F. As-rigid-as-possible shape manipulation. *ACM Trans. Graph. 24*, 3 (2005), 1134–1141.

19. ISHII, H., AND ULLMER, B. Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proc. CHI '97* (1997), ACM, pp. 234–241.

20. KANDA, T., KAMASIMA, M., IMAI, M., ONO, T., SAKAMOTO, D., ISHIGURO, H., AND ANZAI, Y. A humanoid robot that pretends to listen to route guidance from a human. *Auton. Robots 22*, 1 (2007), 87–100.

21. KASSIN, S. M. Heider and Simmel (1944) revisited: causal attribution and the animated film technique. *Review of Personality and Social Psychology 3* (1982), 125–150.

22. LEE, J., AND LEE, K. H. Precomputing avatar behavior from human motion data. In *Proc. SCA '04* (Aire-la-Ville, Switzerland, Switzerland, 2004), Eurographics Association, pp. 79–87.

23. LIEM, M., VISSER, A., AND FROEN, G. A hybrid algorithm for tracking and following people using a robotic dog. In *Proc. HRI '08* (2008), ACM, pp. 185–192.

24. LOCKERD, A., AND BREAZEAL, C. Tutelage and socially guided robot learning. In *Proc. IEEE/RSJ IROS '04* (USA, 2004), vol. 4, IEEE, pp. 3475–3480.

25. MAULSBY, D. L., WITTEN, I. H., AND KITTLITZ, K. A. Metamouse: specifying graphical procedures by example. In *Proc. SIGGRAPH '89* (NY, USA, 1989), ACM, pp. 127–136.

26. NORMAN, D. A. Emotional Design: Why We Love (or Hate) Everyday Things. Basic Books, NY, 2004.

27. OTERO, N., ALISSANDRAKIS, A., DAUTENHAHN, K., NEHANIV, C., SYRDAL, D., AND KOAY, K. Human to robot demonstrations of routine home tasks: exploring the role of the robot's feedback. In *Proc. HRI '08* (NY, 2008), ACM, pp. 177–184.

28. RAFFLE, H. S., PARKES, A. J., AND ISHII, H. Topobo: a constructive assembly system with kinetic memory. In *Proc. CHI '04* (NY, USA, 2004), ACM, pp. 647–654.

29. SCHOLL, B. J., AND TREMOULET, P. D. Perceptual causality and animacy. *Trends in Cognitive Sciences (TRENDS) 4*, 8 (Aug. 2000), 299–309.

30. SHARLIN, E., WATSON, B., KITAMURA, Y., KISHINO, F., AND ITOH, Y. On tangible user interfaces, humans and spatiality. *Personal and Ubiquitous Computing 8*, 5 (2004), 338–346.

31. STRAUSS, A., AND CORBIN, J. *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*. Sage Publications, Thousand Oaks, London, New Delhi, 1998.

32. THOMAS, F., AND JOHNSTON, O. *The Illusion of Life: Disney Animation*, first hyperion ed. ed. Disney Editions (Walt Disney Productions), 1981.

33. THOMAZ, A. L., AND BREAZEAL, C. Teachable robots: Understanding human teaching behavior to build more effective robot learners. *Artificial Intelligence 172* (2008), 716–737.

34. TREMOULET, P. D., AND FELDMAN, J. Perception of animacy from the motion of a single object. *Perception 29*, 8 (May 2000), 943–951.

35. YAMAOKA, F., KANDA, T., ISHIGURO, H., AND HAGITA, N. How close?: model of proximity control for information-presenting robots. In *Proc. HRI '08* (NY, USA, 2008), ACM, ACM, pp. 137–144.

36. YANCO, H. A., AND DRURY, J. L. Classifying human-robot interaction: an updated taxonomy. In Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 2004. SMC '04, The Hague, The Netherlands, October 10–13, 2004 (Oct. 2004), vol. 3, IEEE, pp. 2841–2846.

37. YOUNG, J. E., HAWKINS, R., SHARLIN, E., AND IGARASHI, T. Toward acceptable domestic robots: Applying insights from social psychology. *Inagural Issue of the Int. J. Social Robotics. January 2009 1*, 1 (2009).

38. YOUNG, J. E., IGARASHI, T., AND SHARLIN, E. Puppet master: Designing reactive character behavior by demonstration. In *ACM SIGGRAPH / EG SCA '08*. (Germany, 2008), EG Press, EG Association, pp. 183–191.