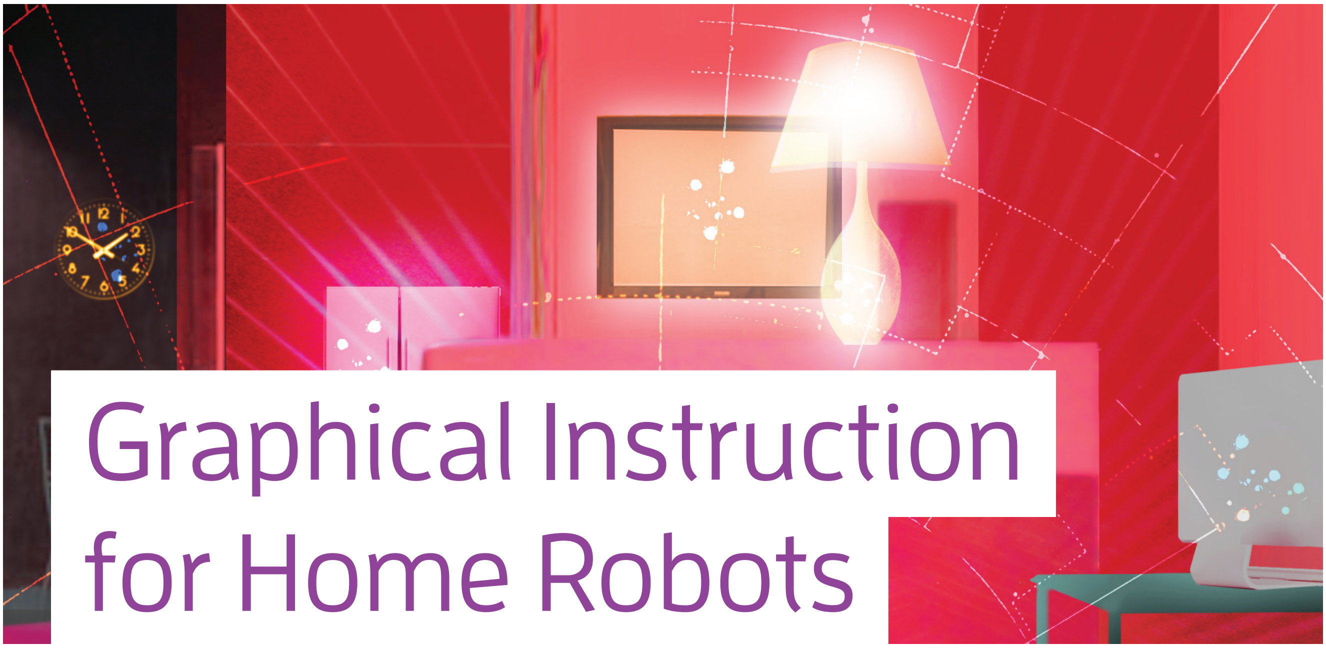


Graphical Instruction for Home Robots

Daisuke Sakamoto, University of Tokyo
Yuta Sugiura, Keio University
Masahiko Inami and Takeo Igarashi, University of Tokyo

Three projects—a cooking robot, a garment-folding robot, and an integrated household task-management interface for managing multiple robots—demonstrate proof-of-concept prototypes showcasing the future of home robots and their interaction design.

Many of us dream of a perfect home robot that does all our household chores. Usually, we expect interaction with this robot to be both verbal and nonverbal. For example, we say what we want done while pointing, and the robot understands what we intend. This multimodal interaction is intuitive for people but can be difficult for computers. Problems can happen with implementation, because even state-of-the-art technology sometimes cannot understand verbal and nonverbal instructions.

Another problem is the difficulty of modifying or editing verbal instructions after the fact. For example, we might say to a robot, "Put this box in front of the door by the evening," but sometime later reconsider and say, "Wait, put it just next to the door by tomorrow." This interaction is simple for users but can be incredibly difficult for computers.

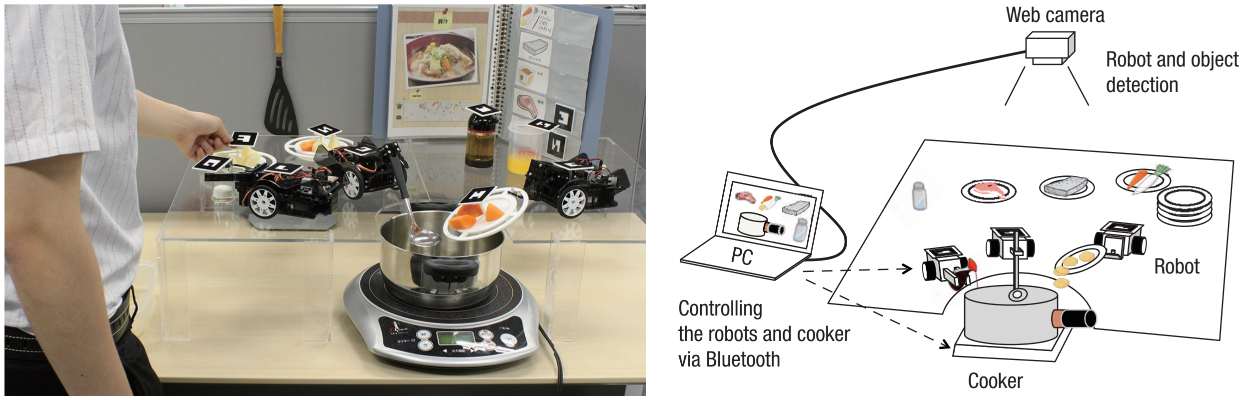
We believe that a GUI is better suited for controlling home robots and dealing with these issues. So, we developed three prototype GUIs for home robot systems. One is for robots that cook, one is for a robot that folds laundry, and one manages groups of home robots.

To create these GUIs and robot systems, we used our Phybots toolkit.[1] It's an all-in-one package to manipulate robots and obtain and manage environmental information to rapidly prototype interaction design for robot systems. It provides a bird's-eye view of the environment and object detection systems with visual markers[2] and optical motion capturing. We abstracted the hardware layer (robot manipulation and sensors) so that we can more quickly develop systems to demonstrate our research.

THE HOME ROBOT AND ITS INTERFACE

The humanoid robot is the archetypal robot; when people say "robot," they usually mean a humanoid robot.

 0018-9162/16/$33.00



(a) (b)

FIGURE 1. The Cooky robot system. (a) An overview snapshot. (b) The system configuration. The user does what the system cannot, such as the cooking preparation. The system does the other tasks, such as placing ingredients in a pot, adjusting the stove's heat, and stirring the pot.

However, our projects use small mobile robots, for two reasons. First, they are smaller than humanoid robots and can be easily stored in the home—for example, in a cabinet. Also, it is more natural to use a GUI to instruct a small mobile robot that does not have a clear mapping of its movements to human-like nonverbal cues.

The combination of a small mobile robot and a GUI makes such robots ideal for housework. People will soon start using them just like home appliances: a cooking robot, a vacuuming robot, and so on. They will also have new home appliances that can move around the open environment—for example, a robot that brings them their coffee every morning or moves items according to their instructions. Other examples might be an automatic bookshelf or cupboard that organizes its contents, a lawn mower that moves to wherever its user points, movable trash bins that collect rubbish around a room as scheduled, and a self-making bed.

Research about GUIs for controlling robots has been conducted frequently in the field of human robot interaction. Such research has targeted mainly extraterrestrial planetary exploration, military missions, undersea operations, search and rescue, and so on.[3,4] Almost all telepresence robots use GUIs for their remote control, and some are Web-based.[5] However, not much research has investigated GUIs for home robots. As we just mentioned, home robots include not only robots that move around the house but also stationary furniture and household appliances. Creating instructions with GUIs for mixed, multidevice robot teams has not yet been sufficiently addressed.

We believe that such GUIs have advantages over multimodal interaction in some cases. These GUIs have four key features. The first is *indirect manipulation*: users simply create instructions on the GUI, and the system understands and reconstructs a task that the robot easily performs. In contrast, in direct manipulation, a user must control all of a robot's actions (for example, moving an arm and opening and closing fingers one by one to grasp a cup).

Second, with *asynchronous operation*, users do not have to monitor the robot while it works. A key advantage of automated home appliances is that they free users to do other things while the appliances are working. The Roomba vacuum cleaner is a good example of this. Asynchronous operation is particularly important to expand the use of home robot systems.

The third feature is *editable instructions*. The most important consideration when designing a GUI is that users can create, modify, save, and load instructions just as easily as if they were manipulating photos in an image-editing application such as Adobe Photoshop. Although considerable research has investigated how to perform task instruction in robot systems, less attention has focused on how to edit instructions after they have been given.

Finally, *timeline-based instruction and interfaces* are popular in animation and video production but not in robotics, especially domestic human–robot interaction. A timeline interface lets users easily edit when a robot should perform a task. One variant of a timeline-based interface is a procedural-based interface; in this case, the order, rather than the specific time, is important.

COOKY

Cooky, our cooking-robot system,[6] does not perform the entire cooking process. Generally, even a state-of-the-art robot system would have difficulty identifying ingredients, cutting them, and boiling or broiling them in an open environment. So, we propose a human–robot collaboration framework in which the user does what the robot cannot.

Cooky uses several small mobile robots on a customized table and a pot on an induction-heating stove (see Figure 1). The main ingredients are on customized plates; the seasonings

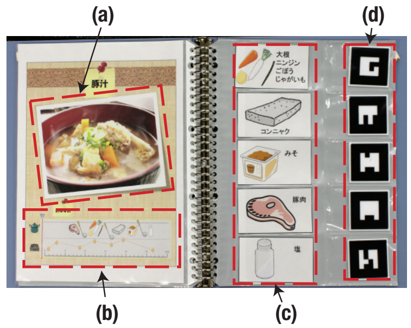


FIGURE 2. The Cooky recipe book. For each recipe, the book contains (a) its name and snapshot, (b) the instructions and timelines for preparing it, (c) the list of ingredients, and (d) each ingredient's visual marker.

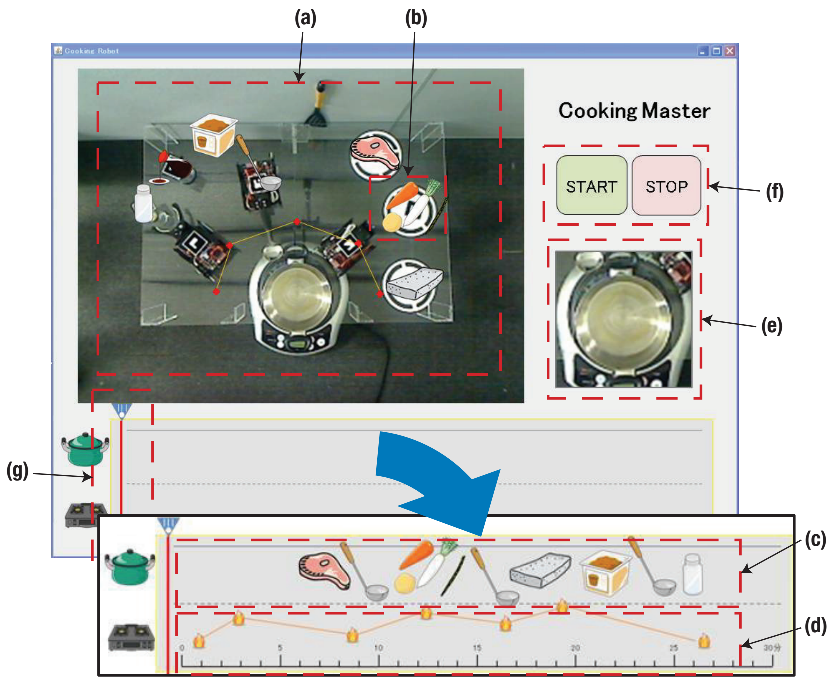


FIGURE 3. The Cooky interface. (a) A live top-down view of the cooking setup. (b) Visual markers overlaid with the corresponding icons. (c) A timeline view showing when to add ingredients. (d) A timeline for temperature control. (e) An enlarged live view of the pot. (f) The control buttons. (g) The progress bar.

are in customized bottles so that the robots can easily identify them. The robots, plates, and bottles have visual markers that let a ceiling camera track them. The robots, stove, and camera are all connected to a PC. The system comes with a special recipe book (see Figure 2) and the GUI (see Figure 3).

First, the user selects a recipe from the recipe book. For each recipe, the book includes its name and snapshot (see Figure 2a), the instructions and timelines for preparing it (Figure 2b), a list of the ingredients (Figure 2c), and each ingredient's visual marker (Figure 2d).

The user cuts the main ingredients according to the instructions, puts each main ingredient on a plate, and places the corresponding visual marker on the plate. The user also pours water in the pot and places it on the cooker. He or she then clicks on the Start button in Cooky's GUI. The system puts the main ingredients and seasonings in the pot one by one, adjusts the heat, and notifies the user when the meal is ready.

The user can also define new recipes. He or she preprocesses the necessary main ingredients and places them and the visual markers on the plates. The system comes with a set of visual markers for common ingredients (onions, carrots, potatoes, beef, pork, and so on), but users can use supplemental visual markers for other ingredients. Then, through the GUI, the user defines the cooking procedure—what ingredients to use and how to adjust the heat. The user sets the timing for adding ingredients by dragging and dropping the corresponding screen icons on the top timeline (see Figure 3c). The user can select multiple copies of an icon to have the system add specific portions of those ingredients to the pot. To control the temperature, the user edits the graph on the bottom timeline (see Figure 3d).

FOLDY

Foldy, our garment-folding robot system, consists of a robot, a stage with a camera on top, and a computer (see Figure 4a).[7] To teach Foldy how to fold a garment, the user places the unfolded garment on the stage and clicks on the Camera button in Foldy's GUI. A camera view then appears on the screen, and the user clicks on the Capture button to capture the garment image. This defines the initial configuration for the folding. Ideally, the robot should be able to start with any arbitrary configuration, but our current implementation starts only with this unfolded configuration.

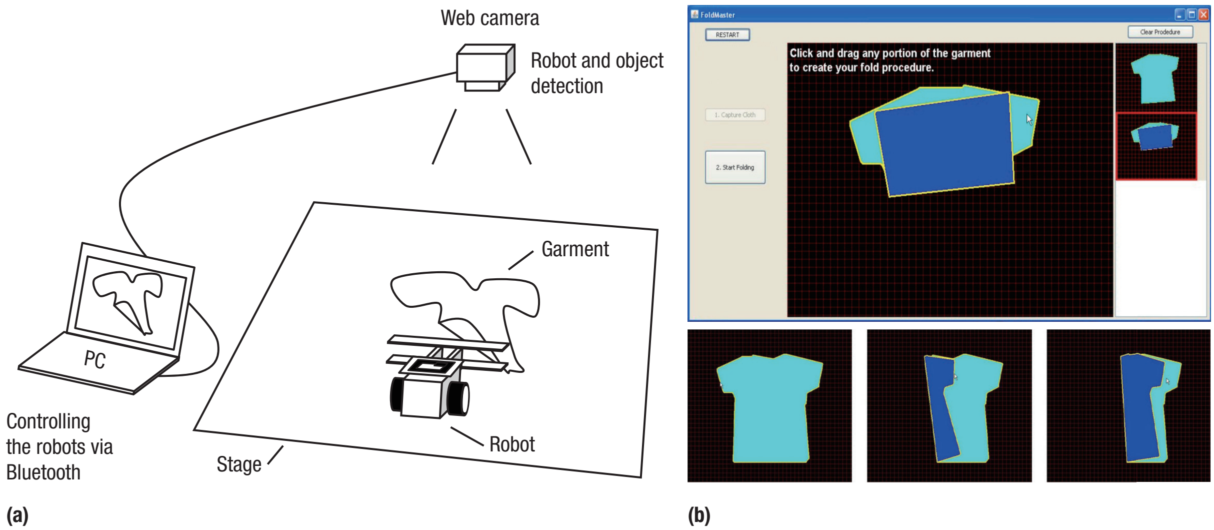


FIGURE 4. The Foldy garment-folding robot system. (a) The system configuration. (b) An overview of the interface (top) and the folding operation in action (bottom). To teach the system, the user folds a virtual captured garment on the computer, using a graphical editor.

The user then folds the virtual garment by clicking on a point on the garment's edge and dragging it to a target location (see Figure 4b). The system continuously provides visual feedback during dragging. A folding step ends when the user releases the mouse button; the result then appears as a snapshot on the history panel (on the right of Figure 4b). The user repeats this procedure until the folding session is done. The user can always go back to a previous step by clicking on the corresponding snapshot in the history panel. The user can also preview the folding process by pressing the Verify button.

Some folding configurations might be physically impossible because of the robot's mechanical constraints, such as when the fold is too small or too large. In those cases, the virtual garment changes color during dragging (see Figure 5). The system asks the user to keep dragging until the garment returns to the default color. If the user releases the mouse when the garment is in an invalid state, the fold fails, and the garment returns to the previous state. This feedback naturally communicates the robot's physical constraints to the user and facilitates the design of a valid folding procedure.

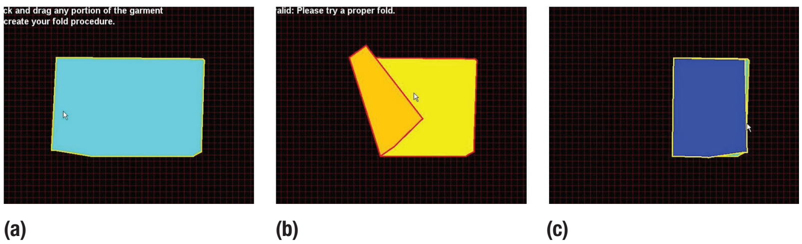


FIGURE 5. Three views of the folding process. (a) Initially, the garment is flat. (b) To notify the user of an invalid fold, Foldy changes the garment's color. (c) The user has completed a valid fold.

After completing a folding session, the user can store it and have the robot perform the actual folding later. The user puts the robot on the stage and clicks on the Start Folding button. The system continuously monitors the robot's progress and adjusts its actions accordingly.

ROBOSHOP

Cooky and Foldy aimed to create instructions for individual robots independently, so they are not really suitable for assigning coordinated action among multiple robots. To address this issue, we present Roboshop, a task-management tool for home robots that employs a graphical editing interface.[8] Roboshop's appearance is similar to typical graphical editing systems, especially image-editing tools such as Photoshop and GIMP (GNU Image Manipulation Program).

To create instructions, the user first selects a tool (for example, the Vacuum tool) from the Housework Toolbox (see Figure 6a). The Property Panel (see Figure 6b) lets users set the tools' properties. For example, users can slow down or speed up the robots' actions or can have robots avoid certain areas.

Next, the user instructs the robot system by sketching on a bird's-eye view of the environment in the SketchPanel (see Figure 6c). For example, the user might specify an area to vacuum by coloring that area. Roboshop employs Rainbow Sketch, which we designed to support multicolored freehand drawings with predesigned optional annotations. It aims to convey clear instructions to robots and convey meaningful task details to users.

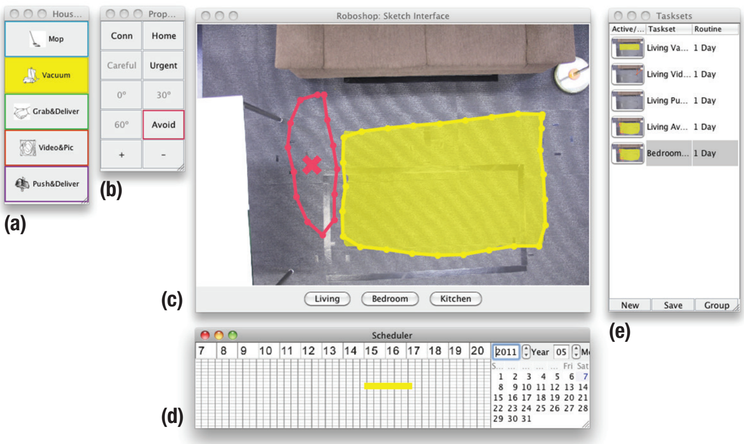


FIGURE 6. The Roboshop task management tool for home robots. (a) The Housework Toolbox. (b) The Property Panel. (c) The SketchPanel. (d) The Scheduler. (e) The Layer Palette. Here, the SketchPanel displays the vacuuming task, showing the area to be vacuumed (highlighted in yellow) and an area to be avoided (indicated by the red border).

Finally, with the Scheduler (see Figure 6d), the user specifies the date and time for the task to start or finish.

Roboshop also provides the Layer Palette for managing multiple tasks in the same room (see Figure 6e) and a grouping tool to extract tasks of interest (not shown in Figure 6). Layers are stored in the Layer Palette; users can restore a layer (for example, a household-task instruction) and edit it anytime. The layered graphical representation gives a quick overview of and access to the rich information tied to the physical environment. Given that the system supports asynchronous heterogeneous robot controls, layers and grouping help users review and manage task sets efficiently.

The GUIs we just presented are not particularly novel; we essentially applied well-known GUI techniques from human–computer interaction to home robot systems. However, as we mentioned before, this has not really been explored in robotics with careful design and evaluation. So, we feel that our three projects are successful first steps in that direction.

We are aware of graphical instruction's limitations; for example, it requires computing devices, whereas multimodal, human-like interactions do not require any devices. Likewise, users need to learn how to use the interface (however, this typically does not take long). Even so, we feel that the benefits of using GUIs outweigh these shortcomings.

We believe that our approach is very general and could contribute to the creation of user interfaces for new appliances that will appear in future smart homes and environments. These appliances might have complicated functions beyond our intellectual capacity and have a completely different kinematic performance compared to current appliances. They might be able to accept instructions by natural conversation.

Nevertheless, graphical instruction will still serve as a default user interface that accepts indirect, asynchronous orders. The graphical representation of appliances and instruction might be more important for interaction with these intelligent appliances because it will become difficult to understand what the appliances are actually thinking.

Overall, combining GUIs with other technologies will open up new research fields. In our case, the combination of a GUI and a home robot has exciting potential applications for human–computer interaction. We believe that GUIs will never die and will still be relevant well into the 21st century.

ACKNOWLEDGMENTS

We performed the projects in this study in collaboration with many researchers. We thank Anusha Withana (Cooky); Hiroki Takahashi, Tabare Akin Gowon, Charith Lasantha Fernando, and Maki Sugimoto (Foldy); and Kexi Liu (Roboshop). We also thank the anonymous reviewers for their constructive feedback.

REFERENCES

1. J. Kato, D. Sakamoto, and T. Igarashi, "Phybots: A Toolkit for Making Robotic Things," *Proc. 2012 Designing Interactive Systems Conf.* (DIS 12), 2012, pp. 248–257.
2. H. Kato and M. Billinghurst, "Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System," *Proc. 2nd IEEE and ACM Int'l Workshop Augmented Reality* (IWAR 99), 1999, pp. 85–94.
3. J.Y.C. Chen, E.C. Haas, and M.J. Barnes, "Human Performance Issues and User Interface Design for Teleoperated Robots," *IEEE Trans. Systems, Man, and Cybernetics, Part C*, vol. 37, no. 6, 2007, pp. 1231–1245.
4. M.A. Goodrich and A.C. Schultz, "Human–Robot Interaction: A Survey," *Foundations and Trends in Human–Computer Interaction*, vol. 1, no. 3, 2007, pp. 203–275.
5. D. Schulz et al., "Web Interfaces for Mobile Robots in Public Places," *IEEE Robotics & Automation Magazine*, vol. 7, no. 1, 2000, pp. 48–56.
6. Y. Sugiura et al., "Cooking with Robots: Designing a Household System Working in Open Environments," *Proc. 2010 SIGCHI Conf. Human Factors in Computing Systems* (CHI 10), 2010, pp. 2427–2430.

7. Y. Sugiura et al., "Graphical Instruction for a Garment Folding Robot," *ACM SIGGRAPH 2009 Emerging Technologies*, 2009, article 12.
8. K. Liu et al., "Roboshop: Multi-layered Sketching Interface for Robot Housework Assignment and Management," *Proc. 2011 SIGCHI Conf. Human Factors in Computing Systems* (CHI 11), 2011, pp. 647–656.

ABOUT THE AUTHORS

DAISUKE SAKAMOTO is a project lecturer in the University of Tokyo's Graduate School of Information Science and Technology. His research interests include human–computer interaction and human–robot interaction. Sakamoto received a PhD in systems information science from Future University Hakodate. Contact him at sakamoto@is.s.u-tokyo.ac.jp.

YUTA SUGIURA is a research associate in Keio University's Faculty of Science and Technology. His research interests include human–computer interaction and user interfaces. Sugiura received a PhD in media design from Keio University. Contact him at sugiura@keio.jp.

MASAHIKO INAMI is a professor in the University of Tokyo's Graduate School of Information Science and Technology. His research interests include physical media and entertainment technologies. Inami received a PhD in engineering from the University of Tokyo. Contact him at inami@inami.info.

TAKEO IGARASHI is a professor in the University of Tokyo's Graduate School of Information Science and Technology. His research interests include user interfaces and computer graphics. Igarashi received a PhD in information engineering from the University of Tokyo. Contact him at takeo@acm.org.