Opportunistic Tangible User Interfaces for Augmented Reality

Steven Henderson, *Student Member*, *IEEE*, and Steven Feiner, *Member*, *IEEE*

Abstract—*Opportunistic Controls* are a class of user interaction techniques that we have developed for augmented reality (AR) applications to support gesturing on, and receiving feedback from, otherwise unused affordances already present in the domain environment. By leveraging characteristics of these affordances to provide passive haptics that ease gesture input, Opportunistic Controls simplify gesture recognition, and provide tangible feedback to the user. In this approach, 3D widgets are tightly coupled with affordances to provide visual feedback and hints about the functionality of the control. For example, a set of buttons can be mapped to existing tactile features on domain objects. We describe examples of Opportunistic Controls that we have designed and implemented using optical marker tracking, combined with appearance-based gesture recognition. We present the results of two user studies. In the first, participants performed a simulated maintenance inspection of an aircraft engine using a set of virtual buttons implemented both as Opportunistic Controls and using simpler passive haptics. Opportunistic Controls allowed participants to complete their tasks significantly faster and were preferred over the baseline technique. In the second, participants proposed and demonstrated user interfaces incorporating Opportunistic Controls for two domains, allowing us to gain additional insights into how user interfaces featuring Opportunistic Controls might be designed.

Index Terms—Haptic I/O, interaction styles, user interfaces, virtual and augmented reality.

1 Introduction

Many domains in which augmented reality (AR) could be applied pose two sets of competing constraints. The first set of constraints limits extraneous head, eye, and hand movements beyond the immediate vicinity of a user's current task. For example, a mechanic servicing an engine may find it impractical (or impossible) to reposition their hands to manipulate any device not currently within reach or sight. Likewise, head and eye movements that cause the mechanic to avert their gaze from the repair area can break context and increase task completion time. The second set of constraints relates to various policies, material properties, and physical space limitations that restrict modifications to the application's environment. For example, safety regulations and a confined repair space might preclude the mechanic from bringing in, or installing, certain interface devices (e.g., portable devices or keypads) that might, otherwise, compensate for limited head, eye, and hand movements.

To support these types of AR scenarios, we have developed a class of interaction techniques that we call Opportunistic Controls, an example of which is shown in Fig. 1. An *Opportunistic Control* (OC) is a tangible user interface [19] that leverages naturally occurring, tactilely interesting, and otherwise unused affordances—properties of an object that determine how it can be used [15], [25]. These affordances serve as tactile landmarks [5] that provide inherent passive haptic feedback [22] for hand gestures and are augmented with overlaid 3D widgets to provide visual feedback. Ideally, OCs are "harvested" from compatible surfaces in the physical task domain of the AR application. As we describe later, certain characteristics of the tactile landmarks are exploited to simplify gesture recognition.

An OC interface enables a user to interact with an AR application by touching naturally occurring surfaces within an application's task environment. For example, a mechanic servicing an engine might use fasteners, such as screws and bolts, located on individually serviced components to display documentation specific to each component. A rotating washer on the same component can be used to page through the documentation or select entries from a list. A grooved surface in the vicinity of the component, such as a door hinge, might map to a virtual spinner used to enter diagnostic data or set various component parameters.

This approach creates a tangible user interface with three distinguishing properties: 1) leveraging otherwise unused and unassociated objects that are already in the task domain as primary user interface components; 2) deliberately exploiting certain features of these objects for passive haptics and hand gesture recognition; and 3) minimizing the need for external user interface artifacts. As we describe below, this generalizes earlier work on passive haptics.

In this paper, which extends our previously published work [16], we begin by examining related works and discussing alternatives to an OC interface. We then provide a formal definition of OCs, and describe a prototype implementation. Next, we describe a performance and acceptance user study involving our prototype and a compared baseline. This is followed by details and results from a second study examining user preferences for the design of OCs.

2 Related Work

There is much previous work on the use of haptic feedback in user interfaces in general and 3D user interfaces in particular.

• *The authors are with the Department of Computer Science, Columbia University, New York, NY 10027-7002.*
E-mail: {henderso, feiner}@cs.columbia.edu.

Manuscript received 17 Feb. 2009; revised 9 June 2009; accepted 7 July 2009; published online 21 July 2009.
Recommended for acceptance by M. Hachet and E. Kruijff.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCGSI-2009-02-0037.
Digital Object Identifier no. 10.1109/TVCG.2009.91.

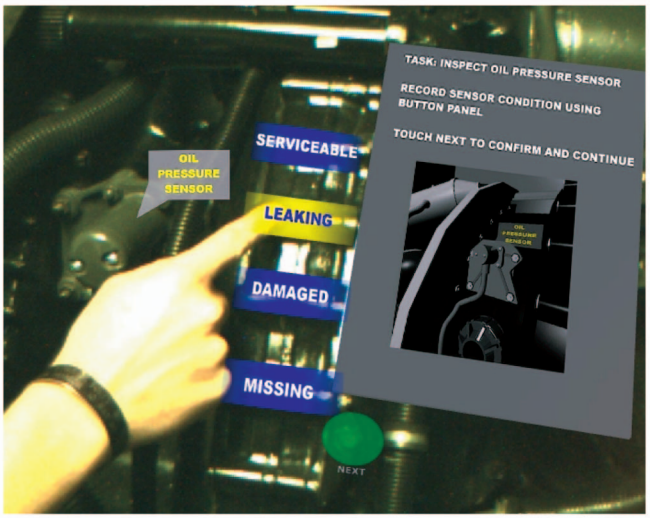


Fig. 1. Opportunistic controls in action: A user manipulates a virtual button while receiving haptic feedback from the raised geometry of the underlying engine housing.

Some of this involves *active haptics* (e.g., [7]), in which active devices, typically using motors, create forces and torques as part of the user interface. Here, we concentrate on previous work on *passive haptics*, in which passive elements in the environment respond to user interaction.

Buxton et al. [8] added a cardboard overlay with cutout holes to a 2D touch tablet, creating a set of separate widgets, each of which could be discriminated through tactile feedback, encouraging eyes-free use. Weimer and Ganapathy [30] positioned a set of 3D virtual buttons operated with a DataGlove to be coplanar with a physical desktop, providing what they called "a natural source of tactile feedback." Later, Hinckley et al. [17] used a ball or doll's head and a small plastic panel, both outfitted with 6DOF trackers as "passive interface props" with which a physician could control an interactive visualization of a patient's head when planning neurosurgery.

Several groups have used tracked hand-held tablets with tracked fingers or styli to provide a supportive mobile surface on which to operate 2D widgets in AR (e.g., Szalavari and Gervautz [28]) or VR (e.g., Lindeman et al. [22]). Lindeman et al. [22] referred to this as "passive haptics" or "passive-haptic feedback." Later work by Insko [18] demonstrated the advantages of passive haptics in virtual environments, positioning styrofoam blocks to coincide with the walls of an otherwise virtual environment. (In fact, one could argue that essentially any immersive virtual environment in which the virtual floor is coplanar with the real floor is using passive haptics.)

Research on *tangible user interfaces* [19] uses a variety of physical artifacts, often tracked or recognized wirelessly, as physical representations of otherwise virtual data and to physicalize otherwise virtual interaction techniques. Fails and Olsen [12] introduced "light widgets" that used optically tracked hand gestures made on everyday surfaces (e.g., the edge of a bed) to control household appliances. While this is an important forerunner of our work on OCs, light widgets do not present any visual feedback to the user (except in a separate application used during camera configuration), do not allow the widget's underlying affordance to move, and do not emphasize the use of differentiated surfaces.

All of this previous work either uses simple naturally occurring surfaces (e.g., [30], [12]) or introduces new objects into the environment, whether simple (e.g., [28]) or more complex (e.g., [17]). In contrast, we are interested in the opportunistic use of objects that not only already exist in a particular task domain, but whose possibly complex surface geometry provides affordances that lend themselves well to certain kinds of interactions. Thus, OCs apply Buxton and colleagues' notion of 2D haptically discriminable widgets to generalize and extend Weimer and Ganapathy's early example of a set of 3D widgets laid out on a single undifferentiated existing surface, without adding additional objects.

3 Alternative User Interfaces

Prior to designing OCs, we considered many alternative interaction techniques involving devices such as keyboards, keypads, and touch screens. If these devices are not readily available within the task domain, they can be added or mobile versions can be used. We rejected these alternatives because of the two sets of competing constraints highlighted in Section 1. Some AR task domains (e.g., aviation maintenance) are not amenable to the introduction of objects that are not indigenous to the domain. Even if mobile devices are made available, they may require a user to shift their hands and eyes away from a specific task. Some require that the user hold them in one hand (e.g., a Handykey Twiddler) or momentarily engage both hands (e.g., a wrist-worn device operated with the other hand).

In contrast, OCs use existing features of the domain environment to provide a suitable tangible user interface. If the user's eyes and hands must remain in a certain area, then affordances within that area may be able to be exploited as part of the user interface. Finally, when the user finishes their task, nothing remains behind that must be maintained, hidden, or removed.

It is important to address potential situations in which the task domain lacks sufficient suitable features for our technique. For example, a user might encounter areas that do not offer enough of the right kind of features to satisfy a task's required number and type of OCs. In these cases, our technique would offer a smooth fallback to conventional passive haptic feedback techniques by binding one or more OCs to undifferentiated available flat surface regions.

4 Definition

We define an OC as the six tuple $\Omega = (\tau, \psi, \alpha, \Gamma, \beta, \rho)$, where:

- τ represents a continuous physical region that bounds the naturally occurring affordance(s) serving as one or more tactile landmarks for hand gestures. This region is specified by a physical model capturing the physical geometry used by the OC.
- ψ is a 3D widget that satisfies the definition and design specifications of Conner et al. [9]. Each instance of ψ consists of a virtual model representing the widget's geometry and an augmented transition network (ATN) specifying behavior.

Fig. 2. Objects that could support button OCs.

Fig. 3. Objects that could support valuator OCs.

- α is a function that maps the encapsulated virtual geometry of the widget (ψ) to the physical geometry of the affordance region (τ). This function dynamically registers the 3D widget's model at the correct location in τ based on the current state of the widget's ATN.
- $\Gamma = \{\gamma_2, \gamma_1, ..., \gamma_n\}$ is the set of visually recognized hand gestures associated with the OC. These gestures share a 3D model space and grammar.
- β represents the functional mapping from the grammar of Γ to the ATN of ψ and defines how an individual widget responds to gesturing.
- ρ is the 3D transformation mapping locations in the model space of Γ to the model space of τ. This transformation is used to detect gesture intersection with the physical geometry of an OC.

It is useful to place our definition of OCs within the broader context of tangible user interfaces. Using Fishkin's taxonomy of tangible user interfaces [14], OCs present a *nearby embodiment* to the user. That is, the output of applications featuring OCs will take place near the primary input device (the OC's physical affordance region, τ). Continuing the classification, each OC presents a *fully realized metaphor* to the user. Given the definition above, the virtual component of the OC (the 3D widget, ψ) is paired to the physical system (the physical affordance region, τ). When the user gestures on an OC, the 3D widget and physical affordance region respond and feel as one control.

5 Prototype

We developed a hardware and software architecture for studying OCs in an indoor laboratory setting. This architecture allowed us to create a prototype implementation that we evaluated by means of the user study described in Section 6.

5.1 Authoring OC Interfaces

We followed a deliberate authoring process when creating our prototype OC interface. This authoring process involves three activities: affordance design, gesture recognition design, and widget design. We expound on each of these activitites in the following sections. In creating our prototype, we executed these activities manually. However, to ensure the practicality of OC interfaces, more research is required to automate the authoring process. Such automated techniques might use real-time vision algorithms to extract interesting affordances and map them to predefined gestures and widgets supporting interface requirements.

5.2 Affordance Design

We experimented with three kinds of affordances in our prototype. The first kind includes unused objects in the environment that tangibly resemble buttons. These objects have physical contours that are easily distinguished by a user's hand. Examples include various fasteners (e.g., screws, bolts, and nuts), raised geometry, small holes, dimples, or the intersection of hard edges, as shown in Fig. 2. OCs based on these types of surfaces support binary gestures in which the OC is activated when the user's hand intersects any part of the button. Here, passive haptic feedback associated with button-based OCs need only provide information about the button's *location* to prove useful (a result demonstrated in the user study). However, certain types of elastic surfaces might provide additional feedback about the state of the button.

The second kind of affordance we explored includes linear or curved *static* surfaces in the environment that could support valuator-based OCs. These include smooth edges, pipes, cords, or natural surfaces, as shown in Fig. 3. Gestures interacting with these types of surfaces require more precise tracking of the user's hand and 3D widgets. More interesting versions of these affordances are characterized by grooves, notches, and other textures that provide discretized feedback to the user as they gesture along the control (e.g., in the spirit of the ridged surfaces designed by Murray-Smith et al. [24] to provide haptic feedback).

The third kind of affordance we studied involves surfaces associated with *movable* objects in the environment. Examples include objects that slide (e.g., the clips on top of a chalkboard shown in Fig. 4, left), objects that bend (e.g., the rubberized tube shown in Fig. 4, center), and objects that rotate (e.g., the disconnected wiring connector shown in Fig. 4, right). These objects allow for richer controls whose underlying physical geometry (τ) moves with the 3D widget (ψ) in response to the user's gestures. However, in practice, movement of these affordances is usually assigned an a priori meaning, which can conflict with the OCs functionality. This limits their potential use in many practical settings.

Throughout this exploration of the space of possible affordances, we adopted the following initial set of heuristic guidelines governing the selection of OCs:

- OCs should avoid desensitizing the user to a function of an overloaded object (e.g., using switches on a control panel for functions outside their design specification). This includes avoiding the use of

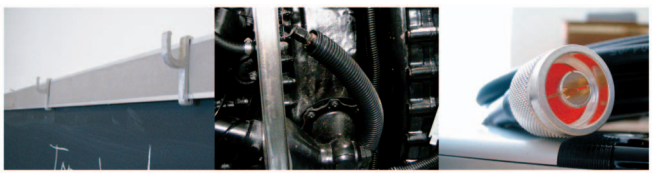


Fig. 4. Objects that could support movable OCs.

objects with strong preconceived purposes and functionality that deviate from the purposes and functionality of the OC. As mentioned above, this guideline especially constrains movable OCs.
- OCs should not endanger the user or desensitize them to surfaces that could prove dangerous outside the context of the OC (e.g., using the tip of a spark plug as a button).
- When applicable, affordances should not overload objects that might become damaged through gesturing (either while the user is manipulating the OC or when the user tries to execute the gesture when the object is assuming its designed purpose).

5.3 Widget Design

We experimented with several designs for 3D widgets (ψ) as part of our prototype development. In each case, we sought to create an appropriate 3D model that matched the particular geometry of the OC. In all cases, we found that increasing the transparency of the widget models was helpful to allow users to partially view the OC's underlying geometry. The transparency also allows the user to partially view any gestures that might be occluded by the 3D widget. The ATNs for each widget are modeled as specified by Conner et al. [9]. This was a trivial process for button-type OCs, and involved slightly more complicated transitions for valuator and movable OCs.

5.4 Gesture Recognition Design

Gesture recognition is performed optically with a single camera mounted overhead with clear line of sight to all OCs in our environment. The camera is tracked by using the ARTag optical marker tracking library [13] to detect a fiducial array within the camera's current frame. We use a separate dedicated camera (as opposed to the cameras supporting the user's display) to free the user from having to look at the OCs. This allows the user to look in another location while gesturing and supports eyes-free interaction.

A separate execution thread analyzes each camera frame for the user's gesture and is implemented in three phases: data reduction, gesture matching, and gesture parsing. In the *data reduction phase*, we build on the appearance-based approach developed by Kjeldsen and Kender [20] to segment each frame to locate the user's hands. The segmentation process first defines the collective gesture model space as one sharing the camera's 2D coordinate system. In doing so, the segmentation algorithm ignores any depth information in the scene. Despite several notable disadvantages discussed in Section 5.5, this relaxation speeds gesture recognition and provides sufficient grammar for our OCs. We next define the physical model for each OC (τ) as a convex polyhedron that generally matches the physical contours of a particular OC. Each polyhedron is defined by 3D points positioned in a common physical interface coordinate system. The algorithm then defines the transformation ρ that enables conversion of coordinates in gesture space (camera coordinates) to and from physical interface coordinates.

This is an important step in the data reduction chain, and a particular advantage afforded by OCs, because it focuses the amount of follow-on image processing required for segmentation. Because the interaction technique is only concerned with gestures that might intersect with specific physical areas, segmentation algorithms can restrict processing to the 2D pixel regions that overlap with each OC's physical model. Moreover, because we track the position and orientation of the camera, ρ is computable in real time by solving for the inverse model-view matrix received from the ARTag library. The algorithm calculates a segmentation window for each OC by using the value of ρ to construct a 2D bounding box encapsulating each OC's physical geometry (Fig. 5a). Each segmentation window is filtered for significant values of the primary color red in the source image's 24-bit RGB color format. When complemented by a controlled lighting environment, this filtering can effectively isolate a user's gesture from other objects in an image and supports a wide range of skin pigmentation. The result is a binary image that represents possible locations of the user's skin touching (or overlapping) each OC's geometry (Fig. 5b).

Fig. 5. Unsegmented (a) and segmented (b) bounding boxes for a set of OCs. Graphics are added in debugging interface. (The user does not see the camera's view.)

The algorithm then executes a connected component analysis for each OC bounding box and assumes the largest component in each is the user's hand, finger, or set of fingers. A high-pass filter is applied to the size of each maximum component to prevent noise from triggering buttons when skin is not present. During this step, the reduced pixel area provided by each OC's segmentation window again helps reduce data processing by limiting the breadth and depth of recursive connected component analysis.

During the *gesture matching* phase of the algorithm, the largest connected component C in each OC is evaluated for the location of point p_h, where p_h approximates the location of the user's fingertip in the connected component. This point is determined by selecting the leftmost point on the highest scan line of C. This approach assumes p_h is the highest leftmost point of the user's gesture in the camera's coordinate system. The algorithm then uses ρ^{-1} to translate the point p_h to the corresponding point t_h in the physical coordinates of the OC (τ). The location of t_h is used to match against the gestures in the OC's gesture set (Γ).

Gesture parsing is accomplished with a finite-state machine for each OC that resembles the ATN of the accompanying 3D widget (ψ). Each state in the finite-state machine represents a command (e.g., "BUTTON_1_DOWN" or "SLIDER_2_UP") in the shared OC grammar associated with Γ and the ATN's transitions are mapped to the OC gestures γ. The gesture algorithm then uses the functional mapping of β to translate the current command to the appropriate state in the corresponding 3D widget ψ.

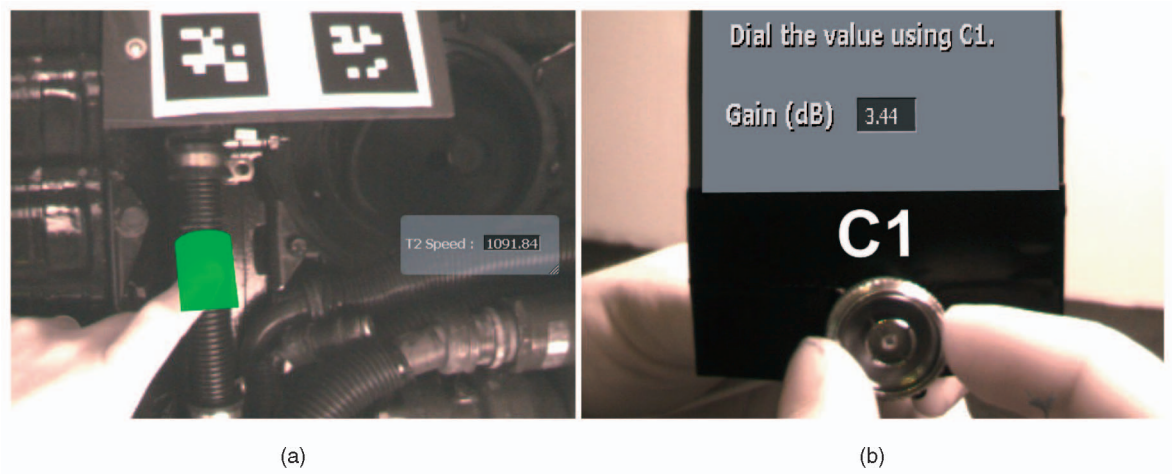


Fig. 6. Additional OCs in our prototype. (a) A valuator-based OC uses the grooves in a wire harness sleeve to provide discrete feedback for a linear slider. (b) A movable OC allows the user to turn the collar of an antenna connector to change a virtual text box value.

5.5 Design Limitations

Our design suffers from several limitations. First, it relies on an optical marker-based tracking scheme to compute the value of ρ. Therefore, markers must be added to the domain environment, contradicting our vision of OCs as not requiring modifications of or additions to the task domain. We believe that it will be possible to build on recent advances in markerless or feature-based tracking [21], [6] to replace our current use of markers. Second, our segmentation algorithm's relaxation of depth information limits the type of interactions one can perform, specifically clutching and hovering. Third, our segmentation algorithm relies on controlled lighting conditions, limiting its practical use in settings outside the laboratory. More work is required to select and incorporate more robust segmentation algorithms into our gesture recognition process. Finally, because each OC's bounding box is segmented separately, the gesture algorithm can produce multiple gestures from multiple OCs. This was a deliberate design decision to support the user gesturing on more than one OC simultaneously (i.e., for multitouch interactions). However, this feature requires more sophisticated program logic to reconcile potentially conflicting gestures. When coupled with our algorithm's lack of depth information, this feature can create situations in which hovering and clutching movements overlap neighboring controls and are erroneously interpreted as active gestures. We discuss this further in the description of our user study.

5.6 Prototype Implementation

We implemented our current prototype using two locally networked computers, one for managing gesture recognition for the OCs, and one for rendering OC widgets as part of a broader AR application testing the OCs in various scenarios. The decision to use two machines resulted in part from concerns about the resource load required to drive a binocular stereo video see-though display, while also supporting hand gesture recognition. Additionally, we are interested in the ability of our software architecture to support scenarios where a single, relatively fixed server and attached cameras could provide gesture recognition to multiple users.

5.6.1 Implemented OCs

Our current implementation features five button-type OCs on a Rolls-Royce Dart 510 turboprop aircraft engine in our laboratory, as shown in Fig. 1. Four of these OCs map to large smooth protrusions on the outside of the engine's compression section and are used to select items in a virtual menu. The fifth button OC maps to a nearby bolt, and is used as a "next" button to navigate between menus. The menu button widgets were modeled to resemble the underlying protrusions, while the "next" button widget is a semitransparent circle.

We also implemented two other types of OCs. One is a valuator-based OC that maps a grooved wiring harness sleeve on the Dart engine to a linear slider (Fig. 6a). This slider is used to control a numeric value recorded in a text box. The other is a rotating OC that maps an antenna connector to a virtual text box (Fig. 6b). As the user rotates the connector's collar, the text box changes value. Fig. 7 shows the engine geometry with and without overlaid graphics.

5.6.2 Gesture Recognition

The gesture tracking algorithm runs on a dedicated Dell M1710 XPS laptop connected to a fixed Point Grey Firefly MV 640×480 resolution color camera tracked by a single ARTag fiducial array mounted near the five button-type OCs (Fig. 8). The gesture recognition application segments the five button-type OCs and parses gestures at 30 frames per second (fps).

5.6.3 Opportunistic Control Application

We implemented a central OC application that integrates all aspects of gesture recognition, rendering, user tracking, and AR task management. This application executes on a PC running Windows XP Professional, with a single NVIDIA Quadro 4500 graphics card. We then attached a custom-built stereo video see-through head-worn display (HWD). This HWD was constructed from a Headplay Visor 800×600 resolution color stereo gaming HWD with two Point Grey Firefly MV 640×480 resolution color cameras mounted to the front and connected to separate IEEE 1394a buses on the PC (Fig. 8).

Tracking is provided by two systems. For the user's head, we used a ceiling-mounted InterSense IS900 6DOF tracker to track a single station mounted on the HWD. Head tracking data is used to position all virtual content, with the exception of the 3D widgets that are part of the OCs. These widgets are positioned using the same optically tracked ARTag fiducial

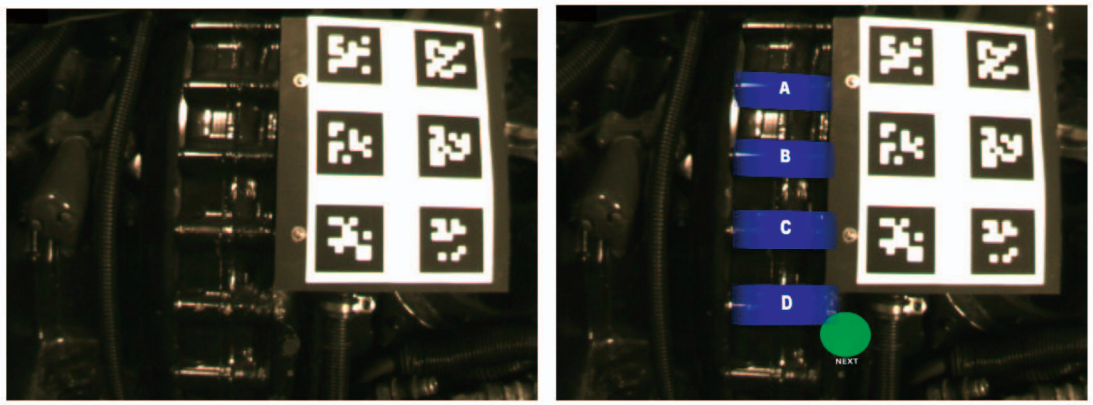


Fig. 7. Prototype interface. (a) Dart 510 Engine without OCs. (b) Dart 510 Engine with OCs.

array used by the gesture recognition camera, sensed with the HWD's left camera. Note that the HWD cameras operate independently of the fixed gesture recognition camera in order to facilitate eyes-free gesture recognition.

The primary AR application software was developed using the Valve Source Engine Software Development Kit. All virtual content in the AR scene is provided by custom game engine models, GUI elements, and other components. Full-resolution stereo video from the two Firefly MV cameras is stretched and displayed on the back buffer and the entire scene is rendered in stereo at 800×600 resolution with an average frame rate of 60 fps, synchronized to the display refresh rate. (Note that the effective video frame rate from the cameras is approximately 25 fps due to the software upscaling from $2 \times 640 \times 480$ to $2 \times 800 \times 600$).

6 OC Performance and Acceptance User Study

We designed a user study to compare the performance and general acceptance of our OC prototype to that of a more standard tangible user interface technique. This study only featured button-based OCs due to ongoing development of our valuator and button-based prototypes at the commencement of the study. Fifteen participants (11 male and 4 female), ages 20-34, were recruited by mass email to the Computer Science students at our university and by flyers distributed throughout the campus, and were paid $\$10$ each. All participants were frequent computer users, but only two had experience with VR or AR techniques or technology. All participants but one identified themselves as right-handed. Eight participants indicated that they required corrective contact lenses or glasses. All participants determined that the separate left and right eye focus adjustments on the Headplay display provided adequate correction.

Fig. 8. A user, wearing a stereo video see-through HWD, manipulates OCs with our prototype. The fixed gesture recognition camera appears at the top of the photograph.

6.1 Baseline Comparison Technique

We selected virtual buttons projected on a single undifferentiated surface as the baseline comparison technique for the study (herein referred to as BL). This technique is similar to the one used by Weimer and Ganapathy [30]. More recent versions optically track the user's fingers, and have proven robust enough for commercialization as "virtual keyboards" [27], [29]. In order to adapt this technique to our prototype, we installed a 60 cm $(\text{width}) \times 78$ cm $(\text{height}) \times 0.3$ cm (thickness) panel of PVC plastic over the top of the part of the Dart engine that we used to implement the OCs described in Section 5.6.1. The panel, shown in Fig. 9, was positioned and curved such that the virtual buttons would appear in the same locations and could use the same tracking and segmentation algorithms as their OC counterparts, but on an undifferentiated surface. The panel was attached with quick release hardware to facilitate a rapid transition between the two techniques during our study.

6.2 Task

Participants were asked to perform a selection task simulating the mechanical inspection of the Rolls-Royce Dart 510 turboprop engine. This selection task, demonstrated in Fig. 10, consisted of matching target text displayed on 3D virtual placards positioned at locations on the engine with a corresponding text entry in a screen-fixed virtual 2D list. The 3D placards are registered to subcomponents of the engine to simulate specific items to be checked during the inspection. Each target text entry corresponds to a technical maintenance failure condition that might be recognized, observed, and recorded by a trained mechanic (e.g., "Broken" or "Cracked"). This target failure condition was randomly

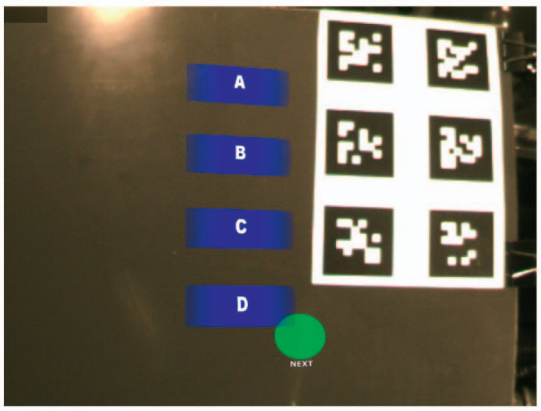


Fig. 9. Baseline comparison technique (BL).

chosen from a list of 32 actual failure codes sampled from an aviation maintenance manual [2].

To successfully complete an individual trial, the user must use virtual buttons to highlight and confirm the target condition in the 2D list. The list contains four positions randomly populated with the target and three incorrect alternative conditions. Participants use four virtual buttons mapped to each position in the list for the highlight step, and confirm the highlighting with a fifth virtual button. Fig. 10 shows an example 3D placard with target text (Figs. 10b and 10c) and the accompanying 2D menu (Figs. 10c, 10d, and 10e), as seen in the HWD.

6.3 Procedure

A within-subject, repeated measures design was used consisting of two techniques (OC and BL) and five inspected locations on the engine. The experiment lasted approximately 60 minutes and was divided into two blocks with a short break between blocks. Each block consisted of all trials for one of the two techniques, and the order was counterbalanced across participants. At the start of the experiment, each participant was shown an instructional video demonstrating the techniques. Before each block, each participant was afforded an opportunity to rehearse the technique using practice trials until they felt comfortable.

The timed portion of the block consisted of 50 trials divided uniformly over five locations on the engine. As shown in Fig. 10, each trial began by populating the virtual environment with a single virtual placard at one of the five randomly chosen locations. Cuing information was first presented to the participant, prompting them to locate and read the target condition displayed on the placard (Figs. 10a and 10b). This portion of the trial was not timed. When the participant positioned and oriented their head so that the placard was under a cross hair in the middle of their field of view, the 2D list appeared and the trial timer started (Fig. 10c). Once the participant used the buttons to highlight and confirm a condition (right or wrong) in the 2D list, the trial ended (Figs. 10d and 10e). The experiment logic then logged the overall completion time, the displayed target condition, and the participant's selection from the list. The block then proceeded to the next trial in repeated fashion until the participant had experienced 10 random target conditions at each of the five locations.

6.4 Hypotheses

Prior to the experiment, we proposed the following hypotheses:

1. OC would be faster than BL, as the differentiable tactile landmarks would reduce homing time and facilitate eyes-free manipulation of buttons.
2. OC would be more accurate than BL, as the tactile landmarks would focus gestures and prevent stray entries.

7 Results

We first filtered our collected data for outliers, which we defined as selection tasks lasting longer than 10 seconds. These outliers accounted for 3.5 percent of all trials, with a total of 23 occurring during the OC block and 29 occurring during the BL block. We then analyzed the remaining data set for completion time, error rate, and subjective ratings, with $\alpha = 0.05$.

7.1 Completion Time Analysis

We applied a 2 (Technique) $\times$ 5 (Location) repeated measure ANOVA on mean selection time from a subset of the outlier free data with our participants as the random variable. This subset included only those trials where the user correctly selected the target condition from the menu (96 percent of our outlier-filtered trials).

Technique had a significant main effect on selection completion times ($F_{(1,28)} = 8.11, p < 0.001$). On average, the OC technique was 16 percent faster (Fig. 11) than the BL baseline technique, which was statistically significant ($t_{(14)} = 4.983, p < 0.001$). This result confirms our first hypothesis. Finally, the interaction of Technique and Location did not have a significant main effect on completion time for the selection task.

7.2 Error Rate Analysis

We applied a 2 (Technique) $\times$ 5 (Location) repeated measure ANOVA on mean errors per trial, with our participants as random variables. However, we failed to identify any

Fig. 10. User study task sequence. (a) Cuing target. (b) Finding target. (c) Reading target. (d) Finding entry. (e) Selecting entry.

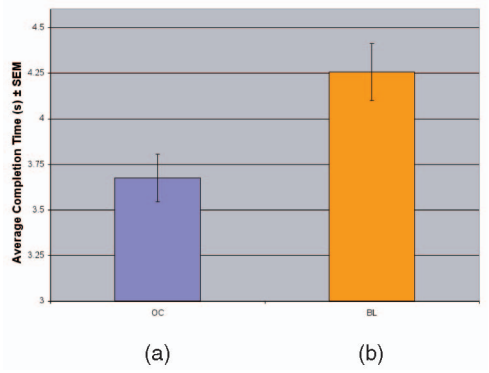


Fig. 11. Average completion times (seconds) for (a) OC and (b) BL. OC was 16 percent faster than BL, which was a significant speedup.

significant effects of technique on error rates ($F_{(1,28)} = 1.94$, $p = 0.185$). The mean error rates were 2.6 errors for OC, and 1.5 errors for BL (Fig. 12) which was not significant. Thus, we fail to confirm our second hypothesis.

We attribute this result to two design shortcomings. First, based on our observations of the experiment and user input, the "next" virtual button was placed too close to the physical protrusion on the engine that was mapped to the virtual button used to select the bottom item in the menu. As a result, the participant's hand gesture could accidentally stray into the segmentation window of this bottom button just prior to activation of the next button. This would erroneously update the participant's selection without allowing time to detect the stray gesture before confirmation. Second, our gesture recognition algorithm does not provide a depth filter. As a result, if the participant's hand hovers over the top of any buttons while transitioning, the algorithm will detect this hovering as button activation. We believe that including depth information in our gesture recognition algorithm and selecting OC affordances more carefully could decrease the number of these errors.

7.3 Subjective Analysis

We asked each participant to complete a postexperiment questionnaire. This questionnaire featured five-point Likert scale questions (where 1 is most negative, 5 is most positive) to evaluate ease of use, satisfaction level, and intuitiveness for each interaction technique. The results from these ratings, shown in Fig. 13 using a technique demonstrated by Bernstein et al. [3], are difficult to generalize given our

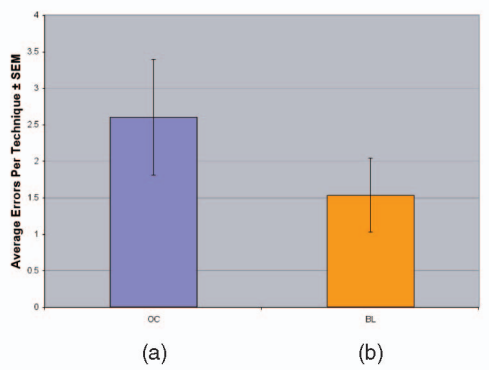


Fig. 12. Average errors per technique for (a) OC and (b) BL. Differences in means were not statistically significant.

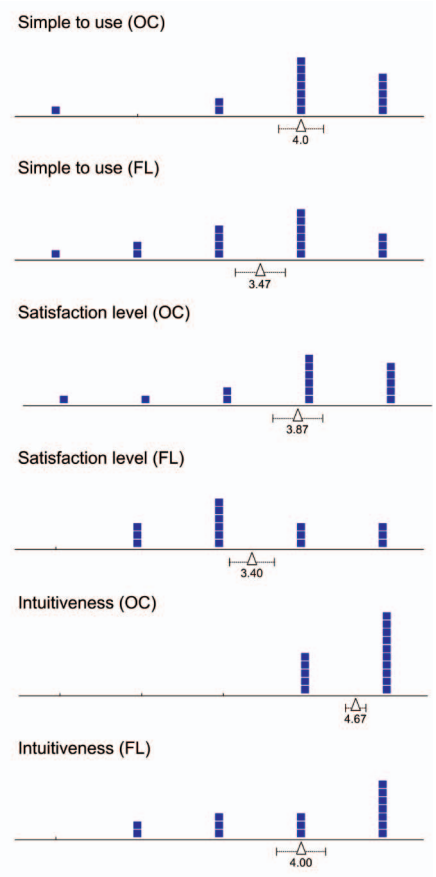


Fig. 13. Survey responses for all participants. Responses are plotted on Likert scales (where 1 is most negative, 5 is most positive) with mean and standard error below the horizontal axes.

small population size and individual rating systems. However, we offer them as interesting indicator of how our technique might be perceived by a larger population. Collectively, the participants rated the OC technique as better than the baseline in terms of ease of use (4.00), satisfaction (3.87), and intuitiveness (4.67). When asked to rank the technique they would rather use to perform the task, 11 of 15 participants selected the OC technique. General participant comments reflected a preference for tactile landmarks to help with homing and feedback. The majority of participants expressed frustration with the top-to-bottom button layout and the inability of the gesture algorithm to distinguish hovering from selection.

We also noticed several interesting behaviors in participants. First, many participants were uncomfortable touching physical parts of the aircraft engine. As one participant recounted, touching the plastic surface of BL felt more familiar than touching louvers and bolts on an engine. Second, several participants used additional passive haptics from the task environment that were not linked to our button OCs to assist in the selection task. These techniques involved incorporating surfaces adjacent to the buttons as homing points between gestures. Third, even though we deliberately did not mention two-handed techniques to the participants, several participants quickly incorporated them into their technique. The fastest recorded completion time was achieved by one such participant.

Additionally, although our user study did not explicitly feature tasks mandating eyes-free interaction, several

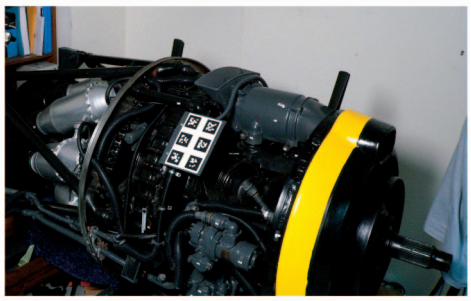


Fig. 14. Aircraft engine inspection domain (MA).

participants tried to select the buttons when they were outside their field of view during both OC and BL trials. Multiple participants commented on how they felt more comfortable attempting eyes-free interaction using OCs as opposed to BL.

8 OC User Interface Design Study

OCs enable a wide range of potential surfaces and objects to be used as interface artifacts. However, based on our experience with the prototype, the design of user interfaces featuring OCs requires careful consideration of how users perceive affordances not typically associated with user interfaces and how best to redirect user thinking to view these affordances as OCs. Additionally, heuristics are needed to determine the best set of affordances from a given domain to meet a particular set of interface requirements.

We designed a user observation study to help answer these questions. Our study rationale was to present participants with 3D interaction tasks that might be encountered when using an AR application within an environment containing a rich set of naturally occurring affordances. Participants would then create hypothetical OCs using any surface or object of their choice, and we would observe the types of surfaces and corresponding gestures that participants selected to accomplish the assigned tasks.

Fifteen additional participants (11 males and 4 females), ages 19-35, were recruited for this study from our university's Computer Science student population, and were paid $15 each. All participants were frequent computer users, and seven reported experience using 3D interfaces. Only one student had participated in our earlier study.

8.1 Task

During the study, each participant was asked to participate in a "Wizard of Oz" session where they created a hypothetical OC-based user interface to perform a series of common 3D interface tasks presented in sample VR and AR applications. These tasks are normally accomplished with common 3D widgets manipulated with traditional input devices. However, in our study, subjects selected any available affordance of their choice and began gesturing to accomplish the particular task while using a "think out loud" protocol to verbalize expected system responses (e.g., "I'm moving the wiring harness to the left to select the wrench with the 3D cursor"). As they gestured, an observer who was out of direct view of the participant used mouse and keyboard inputs to simulate this expected output in our sample applications. This simulated output provided basic visual feedback to the subject.

Fig. 15. Home entertainment domain (HE).

We used the following seven categories from the 3D widget taxonomy proposed by Dachselt and Hinz [11] as the basis for the target interaction tasks presented to each user:

- *3D object selection.* Widgets used to manipulate a 3D cursor to select objects in a scene.
- *3D object manipulation.* Widgets used to rotate and translate a 3D object in a scene.
- *3D scene control.* Widgets used to control the position and orientation of a 3D scene's camera.
- *2D document visualization.* Widgets used to pan and zoom 2D documents in 3D.
- *Discrete valuators.* Widgets modeling a single binary value (e.g., a button).
- *Continuous valuators.* Widgets modeling a continuous range of values (e.g., a slider).
- *Menu selection.* Widgets used to allow selection of items from a list.

We restricted the study to only seven of the most common (based on our experience) members of the taxonomy's 14 widget types in order to limit the scope and duration of the study.

8.2 Procedure

Each participant experienced two application domains that we selected—performing maintenance on an aircraft engine (Fig. 14, herein referred to as MA) and servicing a suite of home entertainment equipment (Fig. 15, herein referred to as HE). We selected these particular domains because they are both rich in tactilely interesting affordances and present experiences that our participants would find unfamiliar (the MA domain) and familiar (the HE domain). For each application domain, participants were given individual tasks from our selected set of common 3D user interaction activities. Both the domain and task orderings were randomized, with the participant experiencing all seven tasks from one domain before proceeding to the next. The

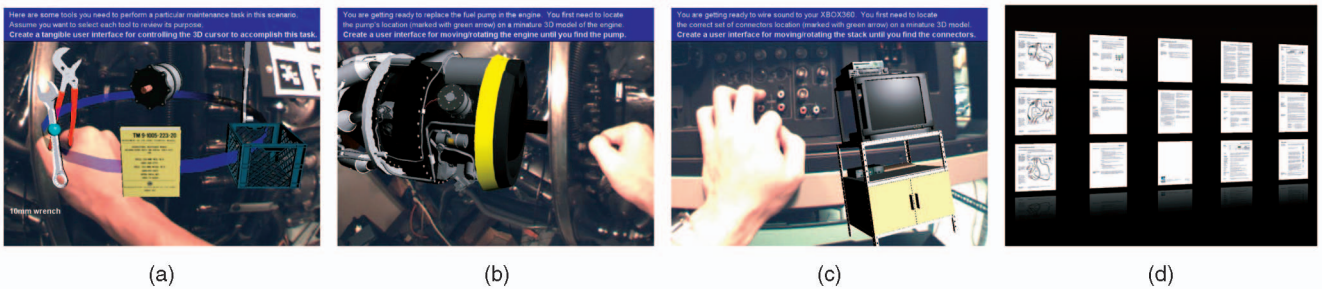


Fig. 16. User observation tasks. (a) 3D object selection and (b) manipulation in MA; (c) 3D object manipulation and (d) 2D document visualization in HE.

entire observation lasted approximately 45 minutes. Individual tasks were presented to the user as part of an unregistered AR application, examples of which are shown in Figs. 16a, 16b, and 16c, with the exception of the 2D document visualization task, which was displayed using the Cooliris [10] application to render 2D images of simulated documentation (Fig. 16d). Fig. 16a shows an example of a participant receiving "Wizard of Oz feedback" from a 3D cursor (positioned by the observer using a keyboard) as the participant selects a virtual wrench by moving an engine hose.

Prior to the observation, each participant signed a consent form, read a one-page set of instructions, and then watched a two-minute introductory video of our connector OC shown in Fig. 6a. The connector shown in this video is not part of either the MA or HE domain. Following this introduction, the user started the first task in the observation. Each task was displayed to the user using an untracked, hand-held Xenearc 700TSV LCD panel with a back-mounted Point Grey Firefly MV camera to generate a video see-through magic lens display [4]. The 3D scenes were rendered at 800×600 resolution using the Goblin XNA augmented reality framework [26]. The 3D models rendered in this study were overlaid on, but not registered with, the physical environment. However, the background provided application context to the user and the device also served as a nonintrusive way to capture what the user was viewing. A stand was provided nearby to hold the display, allowing the user to use both hands for any interaction that they demonstrated.

Data was collected via annotated screen captures taken by the observer through an independent documentation tool not visible to the participant. When the participant indicated a gesture or affordance of interest, the observer snapped a screen shot from the magic lens display, and annotated it with comments from the participant and additional commentary of their own. This documentation system interfaced with the Goblin XNA application via shared memory interprocess communication, and was completely transparent to the participant. This limited extraneous dialogue between the participant and the observer. The documentation tool also tracked the start and finish times of each task.

Once the participant experienced all tasks from both domains, they completed a questionnaire regarding the perceived usefulness of various affordances and a complete description of their recommended user interface for each task in both domains. Participants were encouraged to provide additional hand-drawn sketches, samples of which are shown in Fig. 17. To assist the user in this phase of the study, we provided each with an automatically generated, hypertext-based report that contained all screen captures and observer notes.

8.3 Results

In an attempt to gather insights about user affinities toward possible OC affordances, we examined participant responses to the seven task types using two criteria: OC type and preferred features. We define a participant response as any and all affordances and gestures used to fulfill a particular task. We used the screen captures taken during each session to manually code each participant response as follows: For the OC affordance type criteria, each participant response was examined for presence of affordance types as defined in Section 5.2. In cases where the user invoked multiple affordances to complete a task (e.g., navigating the 3D camera using a valuator for two axes, and two buttons for the third axis), we coded each separately. For the specific preferred feature criteria, we counted and sorted the appearance of specific affordances across all tasks.

8.3.1 Results by OC Affordance Type

Table 1 summarizes the distribution of OC types for each task type across all participant responses.

Study participants selected a plurality of valuator-based affordances, which appeared in 57 percent of tasks in the ME case and 50 percent of tasks in the HE case. It is difficult to

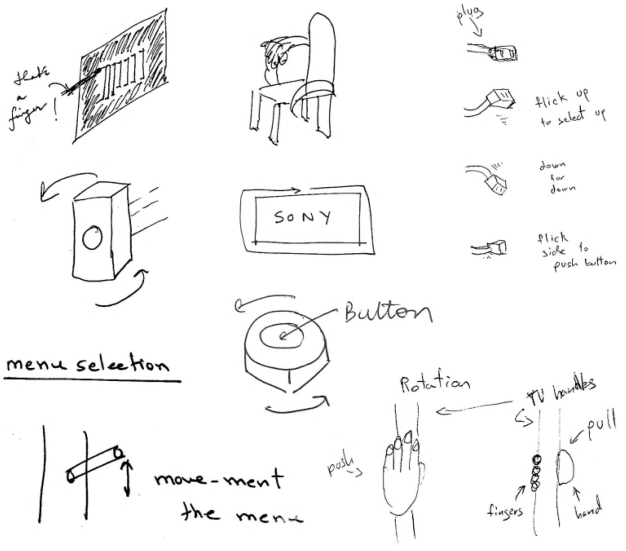


Fig. 17. Selected user concept sketches of proposed OCs.

TABLE 1
Frequency of OC Affordance Types by Task

Task Type	Button	Valuator	Movable
Maintenance Interaction Domain (MA)			
3D object selection	13%	60%	27%
3D object manipulation	7%	87%	20%
3D scene control	13%	73%	27%
2D document visualization	7%	67%	27%
Discrete valuators	67%	20%	13%
Continuous valuators	33%	40%	33%
Menu selection	47%	53%	13%
All Tasks	27%	57%	23%
Home Entertainment Domain (HE)			
3D object selection	47%	40%	20%
3D object manipulation	20%	67%	13%
3D scene control	20%	53%	33%
2D document visualization	20%	67%	13%
Discrete valuators	67%	33%	7%
Continuous valuators	47%	53%	13%
Menu selection	53%	33%	0%
All Tasks	39%	50%	14%

draw conclusions about user affinities from this result due to the influence of task type. The tendency to select valuator-based affordances might easily be the result of participants satisfying device-independent notions about the widgets required to complete a task (e.g., a participant might expect the ability to control volume with a slider). However, the results do provide insights about the general distribution of affordances a typical OC-based application might require.

Examining the results by task type illuminates several interesting findings. In the case of discrete valuators, where one would expect button-based affordances to dominate, our population substituted or incorporated valuator-based affordances 20 percent of the time in the MA scenario, and 33 percent of the time in the HE scenario. In these cases, the participants either fully ignored available button-based OCs (e.g., by tapping the individual striations on a grooved wiring harness sleeve rather than pushing a nearby fastener) or opted to incorporate a valuator as part of the task (e.g., using a scroll gesture on the wiring harness sleeve to first select the desired virtual button, then making a select gesture on the nearby fastener). Examining the continuous valuators task reveals a similar result. In these cases, participants either exclusively used button-based approaches (e.g., changing a spinner's value with up and down buttons, then confirming completion with a third button) or used a valuator-based affordance supported by one or more button-based counterparts. This substitution of valuators and buttons is likely the result of variance in general user interface preferences. For example, when confronted with the task of navigating a menu using a tangible user interface, some users may prefer a user interface that mimics a trackpad, while others might prefer one that resembles the buttons on a keyboard.

Inspection of the results also suggests a general omission of movable OCs. During any given task, a maximum of 33 percent of participants incorporated movable OCs into their interfaces (i.e., when responding to the continuous valuator task in the MA domain). This stands in contrast to preferences for button OCs, which achieved a maximum use by 67 percent of participants during discrete valuator tasks in both domains, and preferences for valuator OCs, which achieved a maximum of 87 percent during 3D object manipulation in the MA domain. We suspect that these results reflect a reluctance on the part of user to modify the environment and the failure of participants to identify movable surfaces due to a lack of knowledge about underlying physical mechanics. Further inspection of the results reveals that participant preferences for OC affordance types were not mutually exclusive. In several instances, particularly for the more complex interaction tasks (e.g., 3D scene control), the participants selected multiple affordances types. In most cases, this was due to a combination of convenience and necessity, where the participant was initially drawn to a particular affordance, later found it lacking, and then added the nearest object. In these instances, it was interesting to watch participants "make their interface work," rather than start over with a more suitable alternative.

Based on these results, one proposed heuristic for the design of OC interfaces, supplementing those of Section 5.2, is to include multiple, possibly redundant, OC types in support of a single interface task. Such an approach might include the ability for a user to dynamically select and configure which controls to use for a task.

8.3.2 Results by Preferred Physical Feature

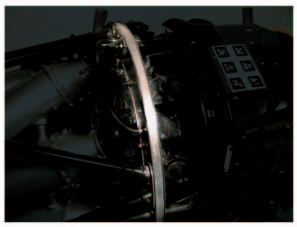
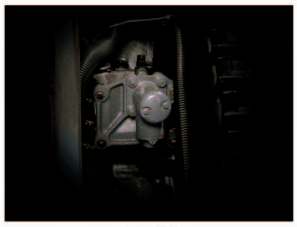
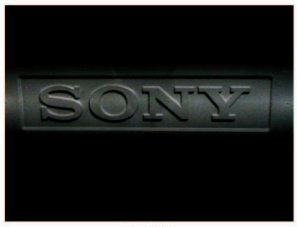
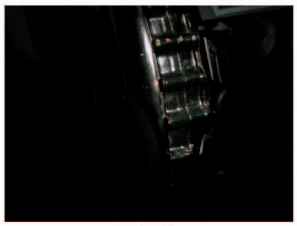
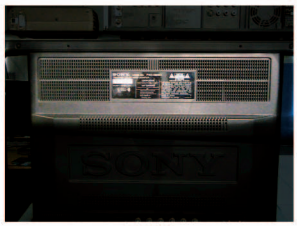
Table 2 lists the top four physical features appearing in participant responses, with each feature manually highlighted in a photograph of the domain. Even though we cannot draw conclusions from these results about what specific features make the best OCs in general, the results do reveal the wide range of affordances users can envision as supporting OCs. Additionally, we noticed that each of the top four selected features in both scenarios were located roughly at eye level and are within arm's reach of where participants stood (unprompted) during the study. This supports the importance of location in the selection of affordances for OCs: The affordance underlying an OC should require minimal physical exertion by the user.

8.4 Additional Findings

We noted several additional findings as a result of this study. First, it was difficult to inspire participants to imagine objects in the MA and HE environments as being components of a computer interface. When participants voiced confusion about what they were supposed to do, we deliberately avoided demonstrating within the MA or HE environments and instead referred the participant to the connector OC video shown prior to the observation. Many participants verbalized their hesitancy to respond with remarks about how they were unfamiliar with the larger objects in the environment (e.g., "I don't know how to hook up a VCR" or "I'm not a mechanically inclined person"). This suggests that even though an object (e.g., the back of a television) might contain objects that are individually perceived as meaningful affordances by the user, context from the area surrounding these affordances can cloud perception. Any implemented OC should make full use of virtual content to help mitigate this effect, possibly even removing or hiding real-world objects that are not part of the OC. (For example, although we did not use any kind of highlighting in this study, techniques such as that used manually in the photographs in Table 2, might be useful to emphasize an important feature in a user interface that employed OCs.)

As part of our questionnaire, we asked participants to suggest other objects found in their daily lives that might be

TABLE 2
Preferred Physical Features (Percent)

MA Domain	HE Domain
 20.0%	 20.0%
 18.0%	 11.0%
 11.0%	 8.0%
 10.0%	 7.0%

used as part of an OC-based interface. Some of the proposed ideas included:

- Using a writing pen to control mobile media players or 3D games.
- Using a chair or sofa arm rest to control a home entertainment system.
- Using the various straps and fasteners on a backpack to control a mobile device (e.g., in the spirit of the wearable communication enabler developed by Mikkonen et al. [23]).
- Using rings worn on fingers to control applications.

9 Conclusions and Future Work

We were pleased that our initial prototype implementation of OC was able to support faster completion times than those of the baseline. Moreover, we were encouraged by the level of enthusiasm for our technique expressed by the participants in our OC performance and acceptance user study. We also believe that minor modifications to our design (e.g., selecting a better arrangement of buttons) could result in a significant improvement over the baseline in error rate performance.

Our current research focus is on adding depth information to the gesture recognition algorithm. We have already implemented a two-camera solution, with one camera mounted parallel to, and just above the dominant plane of our OCs. This allows our system to suspend the segmentation process of the top camera when the user's hand is not seen gesturing at the same depth as our OCs. Initial pilot testing demonstrates improvements in hovering and clutching. We are currently researching more robust options such as a stereo pair of cameras or a depth camera [1]. Other planned improvements in the segmentation process include replacing marker-based tracking with a feature-based approach. We believe many of the same rich features embodied in tactilely interesting OCs could also be leveraged for visual tracking.

We are also interested in developing tools that would allow a user to quickly designate promising looking elements in the environment as OCs. This would require having the user locate a physical object, select a widget type, and specify how the physical object is mapped to the widget. It might even be possible for the system to recognize certain types of features to automatically suggest possible OCs to support the task at hand.

Despite the preliminary insights offered by our OC user interface observation study into how users perceive and interact with OCs, more work is required in this area. Specifically, we would like to determine a set of heuristics that govern which mechanical and free-form topological features fit best with various 3D widgets. These heuristics could be used with the aforementioned tool to help automate the creation of OCs.

In closing, we have presented a class of user interaction techniques for AR applications that support gesturing on, and receiving feedback from, otherwise unused affordances already present in the domain environment. In one user study, a collection of OCs was demonstrated to be faster than a similarly laid out set of controls on an undifferentiated surface. In a second user study, we explored participants' suggested designs for user interfaces that incorporate OC. While not suitable for all user interface scenarios, we believe that OCs may be a good choice for tasks requiring eye and hand focus and which restrict other interaction techniques.

Acknowledgments

This research was funded in part by DAFAFRL Grant FA8650-05-2-6647 and ONR Grant N00014-04-1-0005, and a generous gift from NVIDIA. The authors also thank Bengt-Olaf Schneider of NVIDIA for providing the StereoBLT SDK used to support the display of stereo camera imagery.

References

[1] 3DVSystems, http://www.3dvsystems.com, Feb. 2009.
[2] US Army, "Functional Users Manual for the Army Maintenance Management System-Aviation (pam 738-751)," 1992.
[3] M. Bernstein, J. Shrager, and T. Winograd, "Taskposé: Exploring Fluid Boundaries in an Associative Window Visualization," *Proc. 21st Symp. User Interface Software and Technology (UIST '08)*, pp. 231-234, 2008.

[4] M. Billinghurst, H. Kato, and I. Poupyrev, "The Magicbook—Moving Seamlessly between Reality and Virtuality," *IEEE Computer Graphics and Applications,* vol. 21, no. 3, pp. 6-8, May/June 2001.

[5] G. Blaskó and S. Feiner, "An Interaction System for Watch Computers Using Tactile Guidance and Bidirectional Segmented Strokes," *Proc. Eighth Int'l Symp. Wearable Computers,* pp. 120-123, 2004.

[6] G. Bleser and D. Stricker, "Advanced Tracking through Efficient Image Processing and Visual-Inertial Sensor Fusion," *Proc. IEEE Virtual Reality Conf.,* pp. 137-144, 2008.

[7] F.P. Brooks Jr., M. Ouh-Young, J.J. Batter, and P. Jerome Kilpatrick, "Project GROPE-Haptic Displays for Scientific Visualization," *Proc. 17th Ann. Conf. Computer Graphics and Interactive Techniques,* pp. 177-185, 1990.

[8] W. Buxton, R. Hill, and P. Rowley, "Issues and Techniques in Touch-Sensitive Tablet Input," *Proc. ACM SIGGRAPH Computer Graphics,* vol. 19, no. 3, pp. 215-224, 1985.

[9] B.D. Conner, S.S. Snibbe, K.P. Herndon, D.C. Robbins, R.C. Zeleznik, and A. van Dam, "Three-Dimensional Widgets," *Proc. Symp. Interactive 3D Graphics,* pp. 183-188, 1992.

[10] Cooliris, http://www.cooliris.com, Feb. 2009.

[11] R. Dachselt and M. Hinz, "Three-Dimensional Widgets Revisited—Towards Future Standardization," *Proc. IEEE Virtual Reality Workshop New Directions in 3D User Interfaces (VR),* 2005.

[12] J.A. Fails and D. Olsen Jr., "Light Widgets: Interacting in Every-Day Spaces," *Proc. Seventh Int'l Conf. Intelligent User Interfaces,* pp. 63-69, 2002.

[13] M. Fiala, "ARTag, a Fiducial Marker System Using Digital Techniques," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR '05),* vol. 2, pp. 590-596, 2005.

[14] K.P. Fishkin, "A Taxonomy for and Analysis of Tangible Interfaces," *Personal and Ubiquitous Computing,* vol. 8, no. 5, pp. 347-358, 2004.

[15] J. Gibson, *The Ecological Approach to Visual Perception.* Lawrence Erlbaum Assoc., 1986.

[16] S. Henderson and S. Feiner, "Opportunistic Controls: Leveraging Natural Affordances as Tangible User Interfaces for Augmented Reality," *Proc. Symp. Virtual Reality Software and Technology (VRST '08),* pp. 211-218, 2008.

[17] K. Hinckley, R. Pausch, J.C. Goble, and N.F. Kassell, "Passive Real-World Interface Props for Neurosurgical Visualization," *Proc. ACM SIGCHI Conf. Human Factors in Computing Systems,* pp. 452-458, 1994.

[18] B.E. Insko, "Passive Haptics Significantly Enhances Virtual Environments," PhD dissertation, Univ. of North Carolina, 2001.

[19] H. Ishii and B. Ullmer, "Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms," *Proc. ACM SIGCHI Conf. Human Factors in Computing Systems,* pp. 234-241, 1997.

[20] R. Kjeldsen and J. Kender, "Finding Skin in Color Images," *Proc. Second Int'l Conf. Automatic Face and Gesture Recognition (FG '96),* pp. 312-317, 1996.

[21] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," *Proc. Int'l Symp. Mixed and Augmented Reality (ISMAR '07),* pp. 225-234, 2007.

[22] R.W. Lindeman, J.L. Sibert, and J.K. Hahn, "Hand-Held Windows: Towards Effective 2d Interaction in Immersive Virtual Environments," *Proc. IEEE Virtual Reality Conf.,* pp. 205-212, 1999.

[23] J. Mikkonen, J. Vanhala, A. Reho, and J. Impi, "Reima Smart Shout Concept and Prototype," *Proc. Fifth Int'l Symp. Wearable Computers (ISWC '01),* pp. 174-175, 2001.

[24] R. Murray-Smith, J. Williamson, S. Hughes, and T. Quaade, "Stane: Synthesized Surfaces for Tactile Input," *Proc. ACM SIGCHI Conf. Human Factors in Computing Systems,* pp. 1299-1302, 2008.

[25] D. Norman, *The Psychology of Everyday Things.* Basic Books, 1988.

[26] O. Oda, L.J. Lister, S. White, and S. Feiner, "Developing an Augmented Reality Racing Game," *Proc. Second Int'l Conf. Intelligent Technologies for Interactive Entertainment (INTETAIN '08),* pp. 1-8, 2007.

[27] H. Roeber, J. Bacus, and C. Tomasi, "Typing in Thin Air: The Canesta Projection Keyboard—A New Method of Interaction with Electronic Devices," *Proc. Conf. Human Factors in Computing Systems (CHI '03) Extended Abstracts on Human Factors in Computing Systems,* pp. 712-713, 2003.

[28] Z. Szalavari and M. Gervautz, "The Personal Interaction Panel—A Two-Handed Interface for Augmented Reality," *Computer Graphics Forum,* vol. 16, no. 3, pp. 335-346, 1997.

[29] C. Tomasi, A. Rafii, and I. Torunoglu, "Full-Size Projection Keyboard for Handheld Devices," *Comm. ACM,* vol. 46, no. 7, pp. 70-75, 2003.

[30] D. Weimer and S.K. Ganapathy, "A Synthetic Visual Environment with Hand Gesturing and Voice Input," *Proc. ACM SIGCHI Conf. Human Factors in Computing Systems,* pp. 235-240, 1989.

Steven Henderson is a PhD candidate in the Computer Graphics and User Interfaces Lab at Columbia University where he is researching augmented reality interfaces for procedural tasks. His research interests include 2D and 3D user interfaces, mobile augmented reality, information visualization, intelligent systems, social networks, large-scale database design, and optimization. He is serving as an assistant professor in the United States Military Academy's Department of Systems Engineering, West Point, New York. He is a student member of the IEEE.

Steven Feiner received the PhD degree in computer science from Brown University. He is a professor of computer science at Columbia University, where he directs the Computer Graphics and User Interfaces Lab. His research interests include augmented reality and virtual environments, knowledge-based design of graphics and multimedia, mobile and wearable computing, games, and information visualization. He is coauthor of the well-known text *Computer Graphics: Principles and Practice*, received the ONR Young Investigator Award, and, together with his students, has won Best Paper Awards at the User Interface Software and Technology (UIST) Conference, the Conference on Human Factors in Computing Systems (CHI), and the Symposium on Virtual Reality Software and Technology (VRST). Over the past five years, he has been the general chair or cochair for ACM VRST 2008, the International Conference on Intelligent Technologies for Interactive Entertainment (INTETAIN) 2008, and ACM UIST 2004; the program cochair for the IEEE International Symposium on Wearable Computers (ISWC) 2003; the doctoral symposium chair for ACM UIST 2009; and a member of the steering committees for the IEEE Computer Society Technical Committee on Wearable Information Systems, the IEEE and ACM International Symposium on Mixed and Augmented Reality, and ACM VRST. He is a member of the IEEE.