Design and Validation of Two-Handed Multi-Touch Tabletop Controllers for Robot Teleoperation

Mark Micire[1], Munjal Desai[1], Jill L. Drury[2], Eric McCann[1], Adam Norton[1], Katherine M. Tsui[1], and Holly A. Yanco[1]

[1]University of Massachusetts Lowell, One University Avenue, Lowell, MA 01854
{mmicire, mdesai, emccann, anorton, ktsui, holly}@cs.uml.edu

[2]The MITRE Corporation, 202 Burlington Road, Bedford, MA 01730, jldrury@mitre.org

ABSTRACT

Controlling the movements of mobile robots, including driving the robot through the world and panning the robot's cameras, typically requires many physical joysticks, buttons, and switches. Operators will often employ a technique called "chording" to cope with this situation. Much like a piano player, the operator will simultaneously actuate multiple joysticks and switches with his or her hands to create a combination of complimentary movements. However, these controls are in fixed locations and unable to be reprogrammed easily. Using a Microsoft Surface multi-touch table, we have designed an interface that allows chording and simultaneous multi-handed interaction anywhere that the user wishes to place his or her hands. Taking inspiration from the biomechanics of the human hand, we have created a dynamically resizing, ergonomic, and multi-touch controller (the DREAM Controller). This paper presents the design and testing of this controller with an iRobot ATRV-JR robot.

Author Keywords

Multi-touch, touch screen, tabletop, ergonomic, joystick, controller, finger registration, hand detection, hand registration

ACM Classification Keywords

H.5.2 Information Interfaces and Presentation: User Interfaces - Graphical user interfaces (GUI)

INTRODUCTION

In August 2005, Hurricane Katrina destroyed the majority of the buildings along the coast of Biloxi and Gulfport, Mississippi. Soon thereafter, Florida Regional Task Force Three arrived with a newly designed search robot. During their search, these rescuers were confronted with an apartment building that was so damaged that it was unsafe for anyone, even dogs, to enter. From a safe distance, they controlled a rugged, camera-equipped robot as it explored the building to determine that there were no victims or remains present [15].

The Florida Task Force Three robot operator had hundreds of hours of training using this robot. In fact, most robots for search and rescue (SAR) or explosive ordinance disposal (EOD) require significant amounts of training due to complex interfaces. While there are multiple design approaches for enabling operators to control a robot's movement, sensors, effectors, and lighting, the most common methods involve a large number of joysticks, switches, and dials that each control a particular degree of freedom or function. To activate multiple controls efficiently, operators often employ a technique called "chording" to cope with this situation. In the same way that a piano player will practice to use multiple finger positions to create a harmonic chord, the robot operator will, at times, use multiple fingers on the same hand to manage complex and coordinated movement of the robot.

Depending on the combination and placement of controls and the operator's hand size, chording can result in awkward and fatiguing hand positions. There is a wide variation among hand sizes [5], so a reach that would be comfortable for a 97th percentile male may not even be possible for a 5th percentile female. In the past, there was not much that could be done about this situation: users of these systems would simply have to do the best they could. Now, multi-touch technologies can dynamically adapt interfaces to users' needs instead of the other way around.

Because EOD and SAR place the interface in a safety-critical context in which human lives may hang in the balance, these applications must work reliably and repeatedly. Only recently have technological advancements and economies of scale allowed multi-touch devices to be considered for these challenging domains. To this end, we have been investigating the use of multi-touch technologies for both single robot control and multiple robots in a command and control environment (e.g., [4], [13], and [14]). Our latest interaction design work consists of robot control interfaces that dynamically accommodate for users' different hand sizes and postures. We call our system the dynamically resizing, ergonomic, and multi-touch controller: the "DREAM Controller." This paper documents the DREAM Controller design and two validation experiments: one regarding the underlying registration algo-

IUI'11, February 13–16, 2011, Palo Alto, California, USA.

rithm, and the second exploring first responders' performance using this controller in a simulated search arena.

RELATED WORK

A robot operator control unit display usually includes a video window and status information about the robot [25]. Input devices for interaction with a remote robot system most often are joysticks, keyboards, or mice. Other input devices have also been used, including stylus-based interaction [1, 22, 23]. While the use of speech and gestures have been studied for applications where the robot is collocated with its user [18], they do not transfer well to a remote robot system, as the gesture information is lost without visual contact.

Despite the recent interest in touch technology, single point touch based displays are not new. Their original incarnation was in compact personal digital assistants (PDA) and tablet-based personal computers during the late 1980's and early 1990's. These commercially available devices largely emulated mouse pointer interaction and provided little in the way of further interaction. It is not surprising that there have been few successes in human-robot interaction using these small, often computationally limited, devices.

In Perzanowski et al. [18], a PDA was part of the multi-modal robot control; the user could issue commands from the PDA or select a destination on a map. Keskinpala et al. [10] used a PDA with simple compass-style buttons to drive a robot forward, backward, left, and right. Fong et al. [6] used a PDA to drive a remote robot using waypoint navigation on a video panel or two-axis velocity control using a widget. Beard et al. [1] used a PDA and voice commands to provide high level flight control of an unmanned air vehicle. Skubic et al. [23] also used a PDA to drive a robot; the user sketched a top-down view of the environment and a path for the robot to traverse. A tablet has been used to perform laser laparoscopy using a "what you draw is what you cut" sketch control scheme of a 4-degree of freedom prototype robot [24].

In all of these cases, the use of the stylus or finger touch is limited to mouse-like emulation where the user interaction is limited to pressing buttons, moving sliders, interacting with generated maps, or drawing paths. In most cases, the widgets are standard UI elements where the size and finger occlusions are not optimal. Higher level control is typically expressed in a "go here" command when coupled with a map of the area to be explored.

Recent commercial successes such as the iPhone have brought attention to multi-touch devices. However, this technology actually goes back to the early 1980's when personal computers were beginning their infancy. Bill Buxton [2] provides one of the most insightful and thorough surveys of touch technology through 2008. Multi-touch interaction has great potential to move touch interactions from single point, mouse-like emulation to true bimanual interaction.

Our earlier multi-touch design work attempted to break the mouse-like emulation approach (see [14]). We ported a previously well-studied single robot search and rescue interface

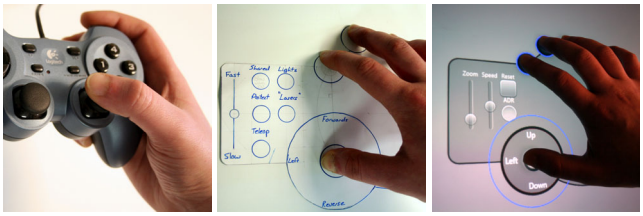


Figure 1. A dual-thumb joystick modeled after the Sony Playstation® controller (left) was used to inspire the design of a paper prototype (center) that was selected for the design of the multi-touch DREAM Controller (right) on the Microsoft Surface.

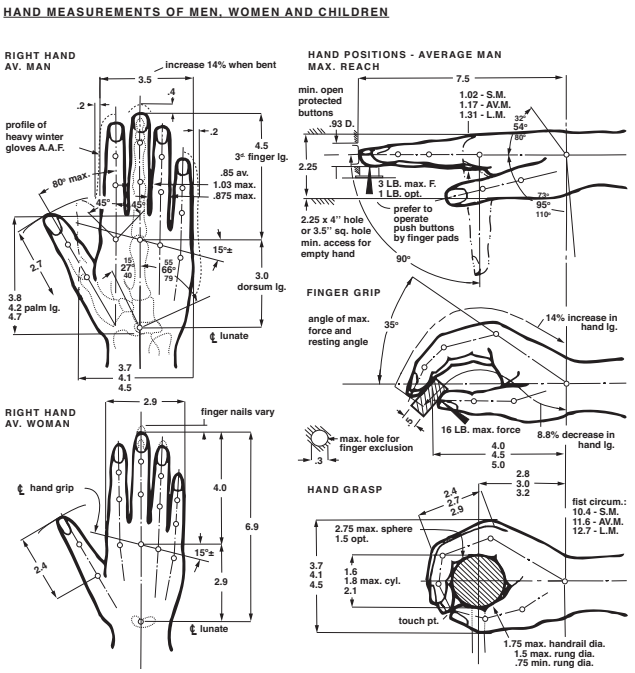
from a traditional physical joystick and non-touchscreen display [11] to a Mitsubishi (now Circle Twelve) DiamondTouch multi-touch table. The physical joystick in the original interface became a virtual joystick widget placed on the lower right side on the DiamondTouch display. Participants in a 2007 study of this interface were enthusiastic about using a multi-touch system to control robots [14]. We found that the DiamondTouch interface did not hinder performance, but also it did not perform significantly better than the original joystick, which indicated that the design could be improved. The new interface and multi-touch controller described in this paper builds on, and is compared to, this earlier work.

DESIGN

One approach to user design is to borrow from outside experiences and interfaces that the user has already encountered. This design method not only helps with ease of learning, but may also exploit muscle memory that the user has developed over time while using the other interfaces. A popular controller paradigm established in the late 1990's by the Sony Playstation® and the Microsoft® XBox® for video games used a dual-thumb joystick design that allowed each of the thumbs to manipulate two degrees of freedom and for various digital buttons and analog pressure sensitive buttons to be incorporated. We surveyed popular first-person games and found that the most common mapping places camera movement (look) on the right thumb and character movement (run and strafe) on the left thumb.

Coupling the ergonomics of the hand with the familiarity of the dual-thumb joystick paradigm, we developed several paper prototypes to determine the feasibility of function and comfort for the user. After several revisions, the multi-touch joystick design (shown in Figure 1) was chosen as the best candidate for further software development. Rather than forcing the close left and right hand positions as in the case of a physical game controller, we decoupled the left and right hands so that the user could maintain all of the functionality of the original dual-thumb joystick design while allowing independent hand movement to any position on the screen.

Our intent in this design was to improve operators' performance, especially by better enabling efficient operation. In particular, we considered ergonomic engineering criteria when driving towards the efficiency goal. We engineered the controller with respect to the resting poses of the human arm, wrist, and hand. The paper prototypes in the early design



HAND DATA	MEN			WOMEN			CHILDREN			
	2.5%tile	50.%tile	97.5%tile	2.5%tile	50.%tile	97.5%tile	6 yr.	8 yr.	11 yr.	14 yr.
hand length	6.8	7.5	8.2	6.2	6.9	7.5	5.1	5.6	6.3	7.0
hand breadth	3.2	3.5	3.8	2.6	2.9	3.1	2.3	2.5	2.8	–
3d finger lg.	4.0	4.5	5.0	3.6	4.0	4.4	2.9	3.2	3.5	4.0
dorsum lg.	2.8	3.0	3.2	2.6	2.9	3.1	2.2	2.4	2.8	3.0
thumb length	2.4	2.7	3.0	2.2	2.4	2.6	1.8	2.0	2.2	2.4

Figure 2. The hand characteristics and registration heuristics were, in part, inspired by Henry Dreyfuss's 1955 book "Designing for People." Reprinted with permission. Courtesy of Allworth Press.

process helped minimize flexion (decrease of angle), extension (increase of angle), and pronation (downward rotation) of the muscles in the wrists and fingers [19]. We also considered that the movements of the thumb, index finger, and little finger have been shown to have much more individualized movement characteristics than the middle or ring fingers [7]. In particular, the movement of the thumb for managing robot movement and camera positioning must be appropriate for accurate and long-term use.

Two sources of information were important in establishing the ergonomic requirements of the DREAM Controller. A wealth of ergonomic information related to gestural interfaces can be found in [16]. In this paper, the authors suggest six key principles of ergonomics: avoid outer positions, avoid repetition, relax muscles, relaxed neutral position is in the middle between outer positions, avoid staying in static position, and avoid internal and external force on joints and stopping body fluids. Each one of these principles was evaluated during the prototyping phase and influenced our design. Another source of anatomical information (and inspiration) was the 1955 book by Henry Dreyfuss [5] containing composite figures of human anatomy as shown in Figure 2 and significant commentary on the ergonomics of the human hand.

HAND AND FINGER REGISTRATION

To systematically create the dynamically sizing control widget, we first needed to robustly recognize and detect individual fingers and handedness. When a finger touches the device, the centroid of the contact is added to a list of point candidates. If there are five or more candidates in the list, the candidate list is passed through a heuristic to determine if those points could contain a subset of five fingers from a single hand. Currently, the heuristic for the hand is a pair-wise evaluation of the candidate points to determine if a subset of those points are within eight inches of each other, which is the maximum possible distance for a human hand based on Dreyfuss's measurements [5] for the largest male hand.

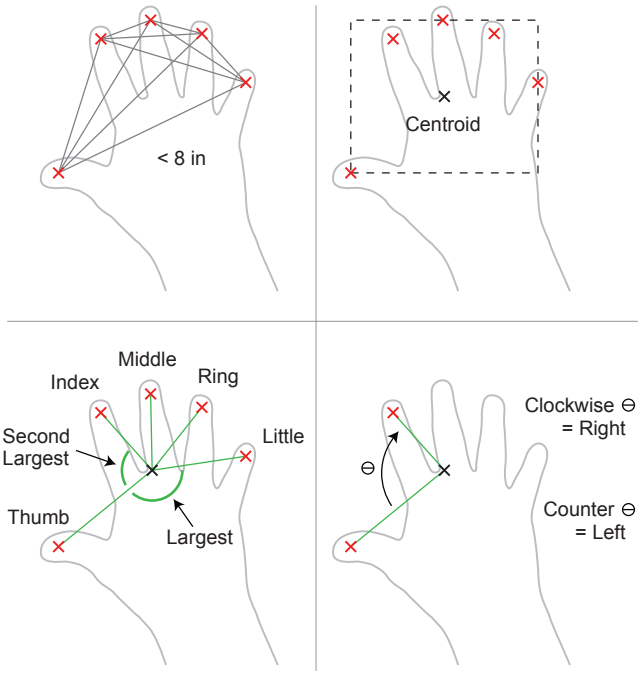


Figure 3. Hand and finger registration is accomplished by first ensuring that the points are within the maximum size for a human hand (top left), then finding the centroid of the bounding box containing all of the points (top right), determining the two largest angles (bottom left), and determining if the angle from the thumb to the index finger is clockwise or counterclockwise (bottom right).

The finger registration algorithm then attempts to figure out which of the five points correspond to specific fingers. To compare the angles between the points, a relatively accurate centroid of the hand needs to be located. A bounding box is created around the five points. The centroid of the box roughly represents a point above the center of the palm, but below the knuckle on the middle finger.

A sorted list of angles between adjacent points and the centroid of the bounding box is then calculated. The largest angle in this list is the angle between the thumb and the little finger. The second largest angle is the angle between the thumb and the index finger. By taking the intersection of these two sets, the algorithm is able to determine the point representing the thumb. The complimentary point on the largest angle is then the little finger and the complimentary point on the second largest is the index finger. Similarly, the complimentary point to the index finger that is not the thumb is the middle finger. The remaining point is the ring finger.

Now that the fingers have been identified, the algorithm can determine if the fingers correspond to a right or left hand. If the angle from the thumb, centroid, and index finger is clockwise, then it is the right hand. If the angle is counterclockwise, then it is the left hand.

With the exception of the initial bounding box to locate the centroid of the hand, it should be noted that the algorithm does not rely on a Cartesian coordinate system and is insensitive to user orientation. The algorithm only uses atan2 and standard arithmetic functions, making it inherently fast and applicable for execution on limited processors. Finally, since the algorithm does not use any of the contact surface area information, it can be used on other multi-touch technologies that only return single pixel touch points.

We tested the hand detection and the finger registration algorithm by collecting data from a wide sampling of people to get a range of hand sizes and natural angles between fingers. Sixty-five people participated in this experiment (21 female, 44 male), each placing their left and right hands in the three positions: on their finger tips in an elevated position with wrist up, on the pads of their fingers in a neutral position, and with their hand maximally stretched. We collected a data set of 390 hand placements on the Microsoft Surface. The algorithm correctly recognized 92% (360 of 390) of the hand placements. It correctly recognized both right hand placements (92%; 179 of 195) and left hand placements (93%; 181 of 195). (See [12] for more details.)

DREAM CONTROLLER FORM AND FUNCTION

The widget algorithm uses the hand geometry and handedness to adjust the size, orientation, and arrangement of the DREAM Controller elements. As shown in Figure 4, there are some key measurements that determine these features. The following description of the widget algorithm will focus on the right hand controller, but the left controller uses the same algorithm mirrored.

First, the angle from the thumb to the index finger determines the orientation of the controller and the button grid. The top right corner of the button grid is placed at the index finger and the lower right corner is placed at the thumb. The width of the grid is determined by the size of the buttons and sliders with the addition of padding for layout and visual balance. The height is the distance between the thumb and the index finger.

A circumcircle that surrounds the triangle created by the thumb, index, and little finger is calculated. This provides an aesthetic visual, showing the users that the controller is tailored to their specific hand size. Programmatically, this circle is also used to protect user elements in lower panels from detecting erroneous events from the finger movement on the controller layer above since the controller can be created and moved to any part of the screen.

A circle pad is placed under the thumb representing analog control for two fully proportional degrees of freedom. Like the Playstation controller, the thumb is then moved up, down, left, and right corresponding to the desired movement. As in

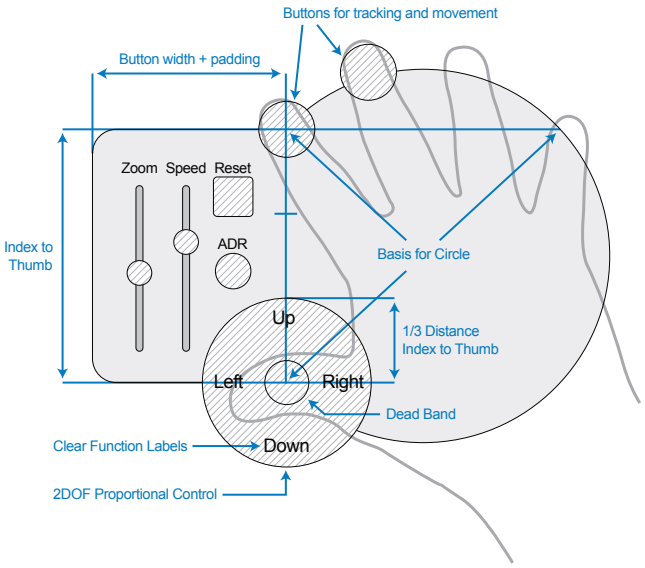


Figure 4. Users have a wide variety of hand characteristics. The DREAM Controller adjusts several parameters to tailor the size, orientation, and position of the controller. This design is intended to maximize the users' comfort and performance.

many first person shooter games, the right hand was used for camera control. The control encircling the thumb rotated the camera's pan-tilt unit (PTU) up, down, left, and right when the thumb was placed at the top, bottom, left and right of the circle respectively. The center of the thumb control represented zero rotational velocity, and this velocity increased as the thumb moved outward from the center. Mixing of the two degrees of freedom was permitted which allowed for fully proportional control of the pan and tilt. A double tap in the thumb area would return the PTU to its origin $(0, 0)$, aiming the camera straight forward and level to the robot chassis.

The panel that extends from the right hand's thumb and index finger included buttons to return the PTU to its origin and toggle Automatic Direction Reversal (ADR) mode. ADR mode allows the user to drive the robot backwards as though they are driving it forwards. These two controls duplicate the functionality of the on-screen gestures, but provide quick local access without the user having to lift his or her hand from the DREAM Controller. Two sliders provide the ability to zoom the camera and adjust the gain on the rotational velocity of the front camera's PTU provided by the thumb.

Also as in first person shooter games, the controller on the left hand is used to move the robot's base. In the thumb control area, movement to the top and bottom moves the robot forward and backwards respectively. The left and right of the circle rotate the chassis via skid steer to the left and right respectively. As with the camera movement, this control surface is fully proportional and permits simultaneous operation of both degrees of freedom.

Three buttons on the left hand panel also control the navigation mode of the robot. These are identical behaviors to the ones provided in our earlier interfaces [11], in which the

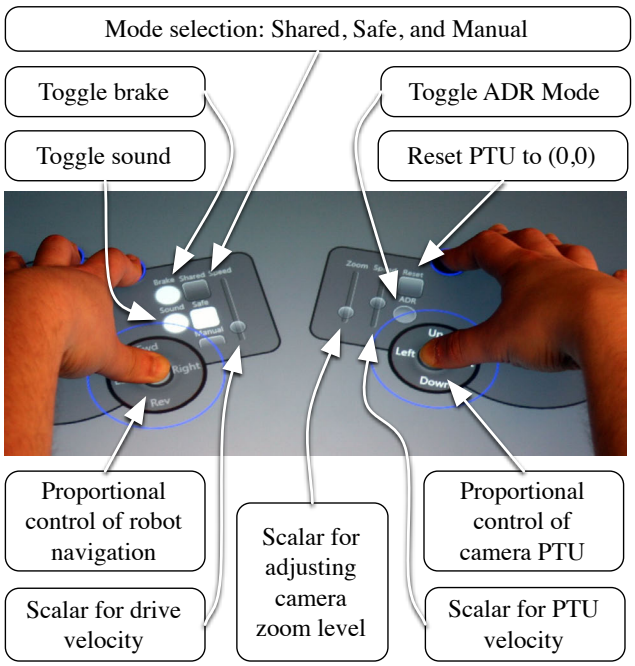


Figure 5. The DREAM Controller, as configured for the ATRV-Jr robot used in this study. Illustration shows the controller functionality and many of the local controls that duplicate on-screen controls.

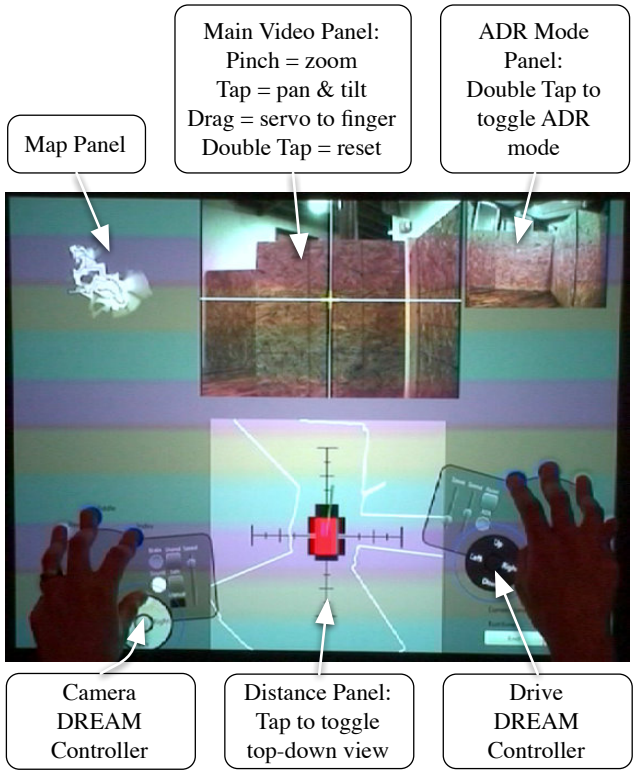


Figure 6. Features and gestures for operating the robot with on-screen controls.

robot could be placed in modes that support different levels of autonomy.

The index and middle finger can be moved to adjust the angle of the controller dynamically if the thumb is not in contact with the joystick. The controller will follow the user's fingers and maintain the orientation of the controller relative to the respective contact points. This freedom of movement is stopped once the thumb makes contact with the two degree of freedom analog control pad. By stopping movement on thumb contact, the controller maintains position and allows the user to relax his or her hand muscles without consequence.

Removal of the DREAM Controller requires the user to simultaneously lift their thumb and their index or middle finger. The user can lift their thumb to stop the motion of the robot and allow repositioning of the controller. Also, at any time while using the controller, the user can lift or lower their ring and little fingers with no effect on operation. We made this design decision for the comfort of the user after noticing this relaxation behavior with the early paper prototypes.

GRAPHICAL USER INTERFACE DESIGN

The Microsoft Surface-based interface uses four panels which evolved through our previous work [11]. The input devices for this interface are the user adjustable DREAM Controllers and on-screen controls for manipulating the robot's functions. Each interaction element is shown in Figure 6.

Main Video Panel: This panel provides the view from the forward facing camera and occupies the center of the display. The video image is augmented by a crosshair that represents the position of the PTU relative to the body of the robot. The crosshair is calibrated so that when the camera is facing straight forward, the crosshairs are centered on the vertical and horizontal view of the camera image. The crosshair coordinate system corresponds to the limits of movement for the PTU, where panning rotation from 0 to 90 degrees leftward will move the crosshair proportionally from the center to the left of the image. This similarly happens for movement to the right, top, and bottom. The crosshairs move proportionally across both degrees of freedom, providing a visual reminder of the rotation of the camera PTU.

If the user taps the main video panel with his or her finger, the PTU will servo to the corresponding (x, y) location with the crosshair providing real-time position feedback. If the user drags the finger, the crosshair and PTU will "chase" the finger and attempt to maintain the crosshair under the finger tip. The PTU motors have to overcome inertia, friction, and other physical properties, so the tracking movement is best-effort and optimized to minimize lag as much as possible. A double-tap on the main video panel resets the PTU to its origin.

Rear Video Panel: The robot has a rear-looking camera that allows the user to see what is behind the robot at any time. This camera's video is placed in the upper right hand corner of the user interface and mirrored on its vertical axis, making it analogous to a rearview mirror in a left-hand drive automobile. A double tap on the rear view toggles ADR mode.

Distance Panel: Directly below the main video panel, a distance panel depicts the robot's chassis with white lines displayed around it representing the distance readings from the sonar sensors and the laser range finder. These distance lines help give the user frame of reference regarding how close the robot is to obstacles that may, or may not, be in the view of the camera. The axes immediately under the robot are shown with 0.25 meter tick marks for reference. With a tap, this panel can be toggled between a perspective view (as in popular GPS navigation systems) and a top down view (as in a paper map).

Map Panel: The upper left corner of the display contains a map that is dynamically generated as the robot travels through the environment. This map, which uses a simultaneous localization and mapping (SLAM)-based algorithm [9], is dynamically generated as the robot is maneuvered around an area. The map shows open space as white and unexplored space as grey. Obstacles are indicated by black lines, the robot's location by a green triangle, and its path by a red line.

Although it was not the intent of the original design, the arrangement of panels left two open areas on each side of the distance and main video panels sufficiently large for two hands. The user was not restricted to only this area, but the open spaces are convenient locations to spawn the DREAM Controller. When the user places five fingers down, the DREAM Controller is activated and when the fingers are lifted, the robot is stopped and the controller is removed.

It should be noted that the DREAM Controller interface omits two panels that were used in our earlier DiamondTouch interface [14] (shown in Figure 7). First, the autonomy mode panel that was previously above the main video was removed and this functionality was moved to three radio buttons on the DREAM Controller. The proportional drive control widget in the lower right was removed along with the speed sliders and brake control. Again, this functionality was moved to the DREAM Controller.

One of the motivations for this change was a minimalist approach to the main window display. Specifically, when the user was not touching the interface, we wanted to maximize the resolution and layout of each of the components of video, range sensing, and mapping. Having the interface cluttered with static buttons and other widgets seemed wasteful when we had determined that the dynamic DREAM Controller could provide this functionality and optimize button placement relative to the users' hands.

EXPERIMENTAL DESIGN

In August 2007, we conducted a study of a static joystick widget on the DiamondTouch (Figure 7) with six participants from the emergency response field [14] (hereafter referred to as the "2007 study"). To compare the DREAM Controller to the earlier DiamondTouch interface design, we repeated the experimental protocol with six new emergency responders. The participants used the DREAM Controller as part of the graphical user interface described above on a Microsoft Surface to drive the robot through a maze to locate victims.

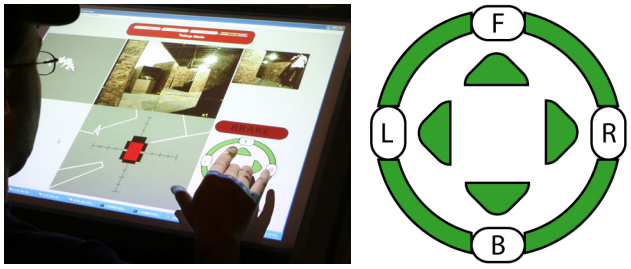


Figure 7. Shoulder (left) and close (right) view of the static 2007 drive control panel, providing control of translation (vertical) and rotation (horizontal).

To the extent that it was feasible, we duplicated the the Reference Test Arenas for Autonomous Mobile Robots at the National Institute of Standards and Technology (NIST) [20] used in the 2007 study. A 2000 square foot maze was built using the same materials. Markers for the ten victims were placed in the same locations as in the 2007 study.

We compared the task performance data of the 2007 study with this study. Since we had two independent participant pools (between subjects), unpaired *t*-tests were used when comparing the interfaces. It would have been ideal to directly compare the DREAM Controller condition with the same participant pool the 2007 study. Unfortunately, the nature of the USAR personnel demographic makes it extremely difficult to capture large numbers of personnel for multiple hours of their limited time. As such, our experiment design attempted to mirror the two testing environments as closely as possible to limit confounders and complications.

Procedure

For consistency between this experiment and the 2007 study, the procedure stayed the same. After signing a consent form, participants filled out a pre-experiment questionnaire requesting demographic information and probing their relevant experience with computers, robots, remote control vehicles, video games, and joysticks. We showed the participants what the robot looks like and then trained them on how to control the robot using the interface. We allowed the participants time to practice using the robot in a location outside the test arena and not within their line of sight so they could become comfortable with remotely moving the robot and the cameras. We then moved the robot to the start location in the arena and asked them to maneuver through the area to find as many victims as possible during a 25-minute period. After task completion, an experimenter asked several post-run questions related to their experience with the interface.

Data collection

We collected four types of data: video, logs, observer notes, and annotated maps. In addition to a video of the robot's progress through the arena, we videotaped over the shoulder of each participant to capture his or her interactions, and we mounted a video recorder pointing down at the multi-touch table. Custom logging software captured each time the participants changed modes, moved the camera, or activated other controls. Two experimenters sat with the participant and

hand-wrote observations. Finally, an experimenter following the robot manually marked its progress through the maze on a run sheet that also provided space to note when and where each bump, scrape, or "e-stop" (emergency halting of the robot) occurred.

Participants

Four men and two women with an average age of 43 years (SD=9.5) participated in this study. All were active members of the USAR community with an average of 8 years (SD=4.5) experience in the field. Disciplines included technical information specialist, communications specialist, canine search, and search. One member was also a search team leader for a wilderness search and rescue group. All had used computers for five years or more, and four assessed their expertise to be at expert or better. All but one had some experience with touch or pen based computing technologies. Half of the participants played video games for an average of 4.5 hours per week (SD=4.4) and half reported that they were pilots. Four of the six had never previously used robots. All participants were right hand dominant in their daily computing activities.

HYPOTHESES

Before our experiment, we hypothesized that the new DREAM Controller on the Surface interface would increase task performance when compared to the static joystick widget on the DiamondTouch interface (H1). That is, we hypothesized that participants would be able to cover more area, find more victims, and create less damage to the environment. We also hypothesized that participants would perform two different control actions simultaneously more often with the Surface interface versus the DiamondTouch interface due to the DREAM Controller design (H2). Finally, we also aimed to qualitatively assess whether participants would be less likely to use the DREAM Controller in unintended ways, as we had seen with the static joystick widget.

RESULTS

We compared the task performance measures related to search coverage and victim identification for the static joystick widget versus DREAM Controller. We also analyzed the participants' use of the DREAM Controller through video coding and logs from the robot and multi-touch device.

Task Performance

Task performance was measured in two ways. First, we measured the distance that the robot traveled in the area, shown in Table 1. This metric was computed slightly differently from the analysis reported in [14] due to the fact that the new DREAM Controller interface allowed several participants to travel so far into the maze that they effectively "lapped" the maze in the allocated time. As such, the distance traveled for both studies is reported including areas that had already been searched. Second, we measured the number of victims found by the operator accounting for overlap in identification. If the participant identified a victim twice or more, the victim only counted as one "find." These two metrics are closely related, since the farther that the robot can travel, the higher the search coverage and probability of detection.

Table 1. Area explored with overlap in the arena in square feet.

Participant	DiamondTouch	Participant	Surface
1	352	7	736
2	320	8	1040
3	304	9	720
4	624	10	1056
5	544	11	192
6	752	12	736
Total	2896	Total	4480
Average	482.67	Average	746.67
SD	185.33	SD	313.15

Table 2. Victims found in the arena.

Participant	DiamondTouch	Participant	Surface
1	5	7	5
2	2	8	7
3	3	9	8
4	7	10	9
5	6	11	3
6	10	12	6
Total	33	Total	38
Average	5.50	Average	6.33
SD	2.88	SD	2.16

We found that participants drove farther using the DREAM Controller interface on the Surface ($\bar{x}$=746.67 square feet, SD=313.15), than with the static joystick widget interface on the DiamondTouch ($\bar{x}$=682.67, SD =288.65) with weak significance (p=0.056, $t(10)$=2.16), using a one-tailed unpaired t-test with unequal variance with α=0.05.

The number of victims found across all participants for the DREAM Controller Surface interface for victim detection showed a 16% increase compared to the static joystick widget DiamondTouch interface (Table 2). However, several factors in this experiment did not allow the data to achieve statistical significance to support the claim that victim detection performance increased. As mentioned above, the overall distance traveled for the Surface interfaces allowed many of the participants to visit areas that they had already searched. Since victims can only be counted once, the participant could no longer receive credit for duplicate victims found. It is not known how many victims that the participants neglected to identify when they realized that they had looped back to their starting location (and several of them did realize they had looped), so this data cannot be easily generated from the raw data set post-hoc. To mitigate this problem in future user testing, the maze needs to be enlarged to exceed the maximum performance expected from the participants.

Based on the experimental design, the use of the DREAM Controller is most likely the contributor to this finding since the control method was the independent variable, with the GUI presentation remaining constant between the two interfaces. Qualitatively, this result is supported by several observations made by the test administrator. One of the ways that participants move quickly through the maze is by focusing on the search task itself and not the robot controls on

Table 3. Comparison in seconds of time spent multi-tasking.

Participant	DiamondTouch	Participant	Surface
1	7	7	23
2	0	8	54
3	1	9	37
4	1	10	51
5	2	11	26
6	2	12	21
Total	13	Total	212
Average	2.2	Average	35.3
SD	2.5	SD	14.4
t-test: 0.00218			

Note: Only 1/3 of the data from Participant 1 in 2007 was preserved, so time was tripled.

the screen. In virtually all cases, by the end of the run, the participants appeared confident that the controller would reliably appear under their fingers in the correct configuration. This interaction was noted since it eliminated the need to look down to their hands and confirm that the controller was configured correctly and ready for input.

Multi-tasking

In Yanco et al. [26], it was found that, on average, robot operators spent 47% of their run time operating the camera to look around the environment. If participants were able to engage in bimanual interaction (i.e., control the camera while simultaneously moving the robot through the environment with the DREAM Controllers), then it follows that the participants would be able to increase the distance traveled and thus find more victims; as in Buxton and Myers [3], we also showed an increase in task performance as described above. We analyzed of the interface logs as well as the videos taken of the participants' hands interacting with the two interfaces to determine how often they were performing multiple tasks at the same time.

Based on the results in Table 3, Hypothesis 2 was clearly supported. There was a significant difference in how well the DREAM Controller enabled participants to activate multiple controls simultaneously compared to the static joystick widget (2.2 seconds vs. 35.3 seconds, $p<0.003$). There are several possible reasons for this difference in performance.

While task efficiency can be improved by activating two controls simultaneously, the static joystick widget design did not allow for the participant to decide where the controls should be placed on the screen. Having two hands and arms on the table rather than one doubled the chance of an arm or hand occluding critical information; a mitigation strategy is to keep one hand well away from the tabletop. In contrast, the DREAM Controller allows sufficient freedom in control (and thus, hand) placement to enable users to view important data while maintaining both hands on the tabletop and directly over the controls. The participants using the DREAM Controller did not have to choose between the efficiency of activating two controls simultaneously and visibility.

Another potential reason for improved multitasking performance is related to the customized sizing of each instantiation of the DREAM Controller. As designed, the controls are within easy reach of the digits on each hand, allowing them to be engaged easily and without spending much time looking at the controls. The amplitude of hand movements is minimal when using the DREAM Controller – literally a twitch of a finger – with the consequence that the energy expenditure needed to activate a control is small. Taken together, these facts can help to explain why participants spent more time using both hands simultaneously to activate controls using the DREAM Controller than using the static joystick widget.

Qualitative Use Characterization

Results from our 2007 study of the DiamondTouch static joystick showed that participants interpreted the joystick widget in a number of different ways, leading to sub-optimal usage [14]. Participants were all trained in the same way, but we saw that each adopted a unique interaction style borrowed from various real-world physical devices. We had intended the interface to evoke joysticks and buttons, but the six participants also demonstrated motions similar to those they would use with mouse track-pads, piano keys, touch-typing, a steering wheel, and sliders. We attributed at least part of the variation to the fact that the design unintentionally included two movement models: proportional velocity (movement and speed being controlled together) and discrete velocity (direction of movement controlled independent of speed). The wide variation in participant behavior told us that we needed to revise the design to use only one movement model and to better align perceived versus actual functionality.

For this study of the DREAM Controller, we examined the video to determine patterns or preferences in how the participants activated the controls. In general, we did not find evidence of participants having trouble interpreting the underlying metaphor or determining how to interact with the controls. Participants kept two hands on the table almost the whole time, using the controls as designers intended, with movements consistent with the metaphor of the DREAM as a flattened physical game controller. The variations in behavior consisted primarily of whether participants activated buttons on the DREAM Controller side panel using the same hand or the opposite hand, and whether they used controls incorporated into the DREAM Controller versus the redundant controls embedded in the display (e.g., touching the video display directly to pan the video camera). Much as touch typists often prefer using "hot key" combinations to avoid the slower process of having to lift their hands from the keyboard and re-home them to the mouse before activating functions, we saw a strong preference among participants to keep their hands on the DREAM Controller nearly constantly.

DISCUSSION AND CONCLUSIONS

The Surface interface using the DREAM Controller allowed participants to rest both of their hands on the multi-touch surface and engage all four degrees of freedom without ergonomic awkwardness or the need for visual positioning of their fingers and thumbs. This hand position allowed them to operate the robot control and the camera at the same time. For

all six participants, we observed continuous periods during their runs in which both hands were fully engaged with the Surface and their thumbs were moving, or ready to move, simultaneously. In our own bodies, we are able to simultaneously walk and look around our surroundings in a natural and intuitive fashion. As such, we should not be surprised to see the participants moving, looking, and increasing their understanding of the environment when the controller makes this possible.

A common and reasonable negative reaction to multi-touch interfaces is based on the objection that high precision control requires direct mechanical feedback to the operators' hands. In the case of classic joystick control, this longstanding belief is embodied in the volumes of ergonomics literature and decades of successful product design. The tension of the spring-loaded gimbals, the debounce of the buttons or triggers, and the shape of the control stick are just a few examples of ways that engineers have tuned the "feel" of the joystick to maximize this sensory return path.

After careful investigation using the DREAM Controller as an existence proof, we believe that the need for direct mechanical feedback is not the only way to achieve high precision control. In fact, we believe that a departure from traditional mechanical input device design in favor of multi-touch interaction will not only maintain or increase performance, but can also significantly improve ergonomics and posture.

One of the reasons that the mechanical feedback becomes so important in traditional joystick design is that the user needs a clear way to conform to the shape of the control surface and also to understand the position of the control mechanism in its various degrees of freedom. When this conformity is sufficiently congruent, the psychology literature calls it an affordance [17]. Properly designed door knobs *afford* the property of being turned and pulled. Buttons on a control panel *afford* the property of being pushed. Switches *afford* the property of being flipped. These elements of design walk the line between engineering and aesthetics that, when executed properly, can become sublime and appear to be the "correct" answer regardless of prior experience or bias.

Once the users' hands have conformed to the device, the spring tension and other mechanical feedback properties allow the user to look away from the joystick and concentrate on the task. The user can trust that the nerves in their hands and arms will feel the position of the joystick and they will not have to repeatedly look at their hands to verify that their input to the system is correct.

Just as our users bring biases and metaphors to new interfaces, we as engineers come to the proverbial design table with preconceived ideas of how we are going to overcome the lack of dimensionality and physical interaction in multi-touch interface design. Unfortunately, when working on a 2D glass surface, visually emulating physical affordances may not be the best approach. Flattening the 3D world to a 2D screen and expecting the same affordances while (by virtue of the device) eliminating the mechanical feedback is, in our opinion, a strategy doomed for failure. In the earlier example of the touch typist or pianist, it is not unexpected that a literal 2D projection of a keyboard or piano on a multi-touch surface would not provide the same performance as the real physical device.

As demonstrated by the DREAM Controller, we believe that a design approach which centers closely around the biomechanical design of the human hand may be an appropriate solution. This focus is not the traditional mechanical design for physical input devices in which the designer attempts to find the most correct design for the largest number of people. Instead, the interface should *conform* to the *user* every time that their hands touch the control surface. Hand sizes, finger lengths, and degrees of dexterity are all variables between individuals. Additionally, all of these properties change as the user fatigues while using the interface for extended periods of time. So, even within a single user experience, there may be multiple optimal interaction configurations to be employed as the interaction progresses.

In a presentation in 2006, Jeff Han provided a very succinct argument for user centered design on multi-touch devices.

> "I cringe at the idea that we are going to introduce a whole new generation of people to computing with this standard mouse-and-pointer interface. . . . there is no reason in this day and age that we should be conforming to a physical device. That leads to bad things like [repetitive stress injury]. We have so much technology nowadays that interfaces should start conforming to us [8]."

We believe that it is the ability to adapt to the users' configurations that will give multi-touch interaction an advantage over traditional mechanical device design. In our DREAM Controller, user-centered interaction is provided by "wrapping" the joystick around the fingers of the individual user. The size of the thumb control is automatically sized based on the size of the users' hands. Buttons, dials, and control surfaces are all tailored specifically for the user's comfort and performance. Even in the case where the user moves his or her hand to a different location on the screen, the DREAM Controller will dynamically track to the new location and position itself underneath the user's fingertips. Because the controller's location is so closely coupled with the operator's physical location, we expect that the DREAM Controller use will not likely cause problems with multiple users unintentionally intruding into another operator's working space. (See the work on territoriality on tabletop workspaces by Scott et al [21].)

While testing the DREAM Controller, we observed participants who had never previously interacted with a multi-touch device controlling four degrees of freedom without looking at their hands during their 25 minute runs. The advantage of our user-centered approach was illustrated through the realization of one of our participants when he explained, "It is taking me a little while to understand that the joystick is going to conform to me and not the other way around. It is a little strange, but I like it." Just as touch typists and pianists take time to trust that their hands and muscle memory will act correctly, we are confident that the user's willingness to trust

that the interface will act correctly in the absence of mechanical feedback will increase over time. When this does occur, we believe that new levels of performance and ergonomic comfort will be the result.

ACKNOWLEDGEMENTS

We would like to thank our participants for their help throughout this study. This research has been funded in part by the Microsoft Corporation and the National Science Foundation (IIS-0546309 and IIS-0415224). All product names, trademarks, and registered trademarks are the property of their respective holders.

REFERENCES

1. R. Beard, D. Kingston, M. Quigley, D. Snyder, R. Christiansen, W. Johnson, T. McLain, and M. Goodrich. Autonomous Vehicle Technologies for Small Fixed Wing UAVs. *AIAA Journal of Aerospace Computing, Information, and Communication*, 2(1):92–108, 2005.
2. W. Buxton. Multi-Touch Systems That I Have Known and Loved. Microsoft Research, 2007. `http://www.billbuxton.com/multitouchOverview.html` (accessed Oct. 2010).
3. W. Buxton and B. Myers. A Study in Two-handed Input. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 321–326, 1986.
4. A. Courtemanche, M. Micire, and H. Yanco. Human-Robot Interaction using a Multi-Touch Display. In *2nd IEEE Tabletop Workshop*, October 2007.
5. H. Dreyfuss. *Designing for People*. Allworth Press, 2003 (Original printing 1955).
6. T. Fong, C. Thorpe, and B. Glass. PdaDriver: A Handheld System for Remote Driving. *IEEE Intl. Conf. on Advanced Robotics*, 2003.
7. C. Hager-Ross and M. Schieber. Quantifying the Independence of Human Finger Movements: Comparisons of Digits, Hands, and Movement Frequencies. *J. of Neuroscience*, 20(22):8542, 2000.
8. J. Y. Han. Jeff Han Demos his Breakthrough Touchscreen. Presented at TED, Feb 2006. `http://www.ted.com/talks/jeff_han_demos_his_breakthrough_touchscreen.html` (accessed Oct. 2010).
9. A. Howard. Multi-robot Simultaneous Localization and Mapping using Particle Filters. *Intl. J. of Robotics Research*, 25(12):1243–1256, 2006.
10. H. Keskinpala, J. Adams, and K. Kawamura. PDA-based Human-Robotic Interface. *IEEE Intl. Conf. on Systems, Man and Cybernetics*, 4, 2003.
11. B. Keyes, M. Micire, J. L. Drury, and H. A. Yanco. Improving Human-Robot Interaction through Interface Evolution. *Human-Robot Interaction*, 2010. Daisuke Chugo (Ed.).
12. M. Micire. *Multi-Touch Interaction for Robot Command and Control*. PhD Thesis, University of Massachusetts Lowell, December 2010.
13. M. Micire, M. Desai, A. Courtemanche, K. Tsui, and H. Yanco. Analysis of Natural Gestures for Controlling Robot Teams on Multi-touch Tabletop Surfaces. In *ACM Intl. Conf. on Interactive Tabletops and Surfaces*, 2009.
14. M. Micire, J. Drury, B. Keyes, and H. Yanco. Multi-Touch Interaction for Robot Control. In *Proc. of the Intl. Conf. on Intelligent User Interfaces*, 2009.
15. M. J. Micire. Evolution and Field Performance of a Rescue Robot. In *J. of Field Robotics*, 2007.
16. M. Nielsen, M. Störring, T. Moeslund, and E. Granum. A Procedure for Developing Intuitive and Ergonomic Gesture Interfaces for Man-machine Interaction. Technical Report CVMT 03-01, Aalborg Univ., 2003. `www.cvmt.dk/~fgnet/docs/fgnet_techreport.pdf`.
17. D. Norman. *The Design of Everyday Things*. Basic Books, New York, N.Y., 1988.
18. D. Perzanowski, A. Schultz, W. Adams, E. Marsh, and M. Bugajska. Building a Multimodal Human-Robot Interface. *Intelligent Systems*, 16(1):16–21, 2001.
19. D. Saffer. *Designing Gestural Interfaces: Touchscreens and Interactive Devices*. O'Reilly Media, 2008.
20. S. Schipani and E. Messina. Maze Hypothesis Development in Assessing Robot Performance During Teleoperation. *National Institute of Standards and Technology NISTIR 7443*, September 2007.
21. S. Scott, M. Sheelagh, T. Carpendale, and K. Inkpen. Territoriality in Collaborative Tabletop Workspaces. In *Proc. of the 2004 ACM Conf. on Computer Supported Cooperative Work*, pages 294–303. ACM, 2004.
22. M. Skubic, C. Bailey, and G. Chronis. A Sketch Interface for Mobile Robots. In *Proc. of the IEEE Intl. Conf. on Systems, Man and Cybernetics*, 2003.
23. M. Skubic, S. Blisard, A. Carle, and P. Matsakis. Hand-Drawn Maps for Robot Navigation. In *Proc. of the AAAI Spring Symp. on Sketch Understanding*, 2002.
24. H. Tang, H. Van Brussel, D. Reynaerts, J. Vander Sloten, and P. Koninckx. A Laparoscopic Robot with Intuitive Interface for Gynecological Laser Laparoscopy. *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2, 2003.
25. H. Yanco and J. Drury. Rescuing Interfaces: A Multi-Year Study of Human-Robot Interaction at the AAAI Robot Rescue Competition. In *Autonomous Robots*, volume 22(4), pages 333–352, May 2006.
26. H. A. Yanco and J. L. Drury. "Where Am I?" Acquiring Situation Awareness Using a Remote Robot Platform. In *Proc. of the IEEE Conf. on Systems, Man and Cybernetics*, October 2004.