Evaluation of Touch Control Interface for the Smart Mobile Furniture

R. Maskeliunas, V. Raudonis, P. Lengvenis[1]
[1]*Information Technology Development Institute, Kaunas University of Technology*
Studentu St. 48A-303, Kaunas, Lithuania
rytis.maskeliunas@ktu.lt

***Abstract*—The paper describes our implementation and experimental evaluation of a touch surface control algorithms developed for the smart mobile furniture. We begin the article by presenting the working principle of touch surface, followed by the descriptions of proposed eyes-free control algorithms aimed to avoid random touches. The experimental evolution and analysis shown the problems affiliated with the usage of standard hardware by manipulating it by nose and foot finger (compared vs a hand finger), and allowed the initial determination of recognition accuracy, performance (in time) and overall user rating.**

***Index Terms*—Multimodal interface, human computer interface, smart furniture.**

I. INTRODUCTION

The development of smart mobile furniture and additional intelligent products is influenced by hi-tech advancement leading to smaller and more advanced electrical devices [1]. Enhancing of everyday objects with technology can mean the provision of comfort and support. At a certain point the smart mobile furniture will only limited by the mechanical boundaries of their construction, as well as the unlocking of the technological potential. The communication between devices will lead to the creation of intelligent environments greatly improving the living conditions and providing a social integration solutions.

The paper is further organized in three sections. In the second section the application and the working principle of touch surface is presented, followed by the descriptions of proposed eyes-free control algorithms aimed to avoid random touches as well as the experimental setup and investigation of performance and recognition accuracy using both methods.

II. APPLICATIONS AND WORKING PRINCIPLE

Most mobility devices are normally controlled with wheels, joysticks, keyboards, and other similar hardware devices. However, with advances in technologies, variations of user interfaces for this purpose have become wider. Touch usage can range from simple multimodal HCIs [2] to advance force displays [3], or controlling a robot via touch surface [4]. A touch surface in principle can be used to record any control-gesture drawn with the finger [5] or to design a control panel interior trim component [6]. The development and evaluation of a multimodal touchpad with audio, tactile and visual feedback for in-vehicle applications showed [7] allowed reaching a higher level of performance compared to the typical rotary push-button interfaces. Touch surface can also serve well for the disabled, as in [8] where control of communication flow was implemented via touchpad circuitry. Multimodal interfaces such as combination of laser pointer and a touch screen interface for assistant robot [9] can help disabled person to lift objects from the ground.

A typical modern touch surface uses capacitive surfaces. According to Symantics [10] two electrical conductors are placed flat against each other separated by a thin insulator forming an electrical capacitor by the finger's touch. The position of the cursor's *X* and *Y* coordinates are measured by calculating capacitance of each conductor. Approximate finger pressure *P* is calculated measuring the total amount of capacitance, so the harder the user presses down, the more the finger flattens, the larger contact area leads to larger total capacitance.

A variety of "filtering" algorithms allow converting the raw *X* and *Y* computations into smooth motion even when some disturbances are introduced. A Symantics offers a windowed average algorithm to compute each filtered coordinate value averaging last two not filtered. If *N* is the unfiltered finger position and X_{new} is the result of the filtering operation then

$$X_{new} = \frac{(N_{new}+N_{previous})}{2}, \quad (1)$$

where N_{new} is the most recent finger position, $N_{previous}$ is the previous finger position. Alternatively

$$X_{new} = \frac{(N_{new}+X_{previoua})}{2}. \quad (2)$$

If there is no value available for $N_{previous}$ or $X_{previous}$ (assuming there was one or no touches before)

$$X_{first\ value} = N_{first\ value}. \quad (3)$$

Windowed average filter can be used for smoothing, averaging three or more recent N values or use a weighted average

Manuscript received March 10, 2012; accepted May 12, 2012.
This research was funded by a grant (No. MIP-037/2011) from the Research Council of Lithuania.

$$\frac{1}{4} N_{new} + \frac{3}{4} X_{previous}. \quad (4)$$

A swipe action itself can be described like so. In idle mode there is no pressure and P is 0. If P increased beyond set threshold, then it is assumed that someone touched the surface and X, Y positions started to change, till the value of P returns to 0. The changes of coordinate motion ΔX and ΔY are translated into cursor motions:

$$\Delta X = S_T \times (X_{new} + X_{previous}), \quad (5)$$
$$\Delta Y = S_T \times (Y_{new} + Y_{previous}), \quad (6)$$
$$C_{Xnew} = C_{Xprevious} + (A \times \Delta X), \quad (7)$$
$$C_{Ynew} = C_{Yprevious} + (A \times \Delta Y), \quad (8)$$

where C is the position of cursor; S_T and S_C are the speed values, A is acceleration set in mouse drivers by setting an acceleration value.

A click gesture is described an action of change of pressure P beyond the threshold and return to 0 with little or no difference in X, Y. Fig. 1 illustrates click and drag detection. The P value and the state of the left "button" (click) are plotted against time as the user executes first a simple touch and drag gesture. On the top the jumps in pressure indicates the actions of touch, on the button – the actions of click and drag.

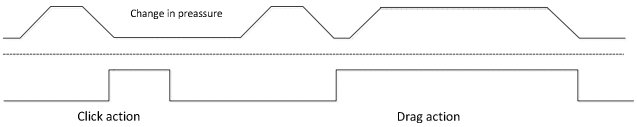

Fig. 1. Tap and drag actions [10].

III. CONTROL ALGORITHMS

Two different proprietary eyes-free control software algorithms were developed to prevent the accidental touch effect (Fig. 2).

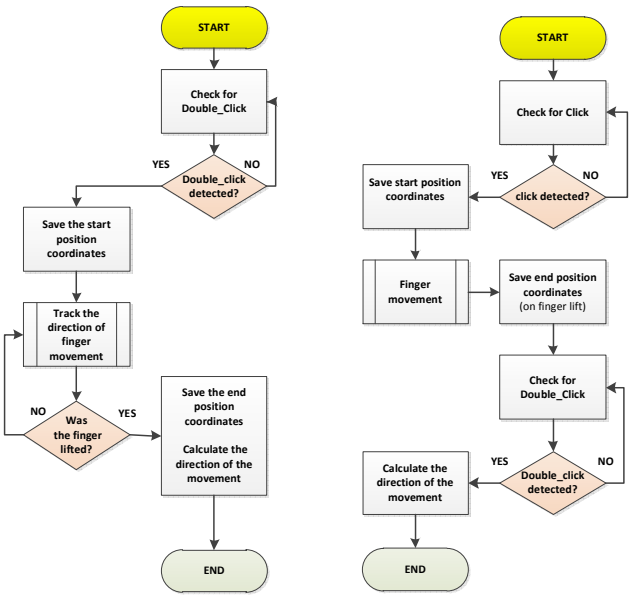

Fig. 2. Control Algorithms ("Double Click and Swipe" on the left; "Click, Swipe and Double Click" on the right).

The result of both of them was the direction vectors use to issue a directional control for our smart mobility vehicle. Both algorithms were realized in .NET using standard tools and function to allow maximum portability and compatibility. Both were integrated into a separate control programs and not on the driver level (not to impact the usability of other system services).

The first algorithm was designed to check for the same spot double click to determine if this is not a random touch, then to calculate a swipe motion while touching the surface. The initial X_{new}, Y_{new} coordinates are registered when a user double clicks the pad, and the final coordinates X_{final}, Y_{final} are registered when $P = 0$ (user lifts a finger) The end result (the directional vector) is calculated from the gathered data.

The second algorithm works somewhat differently. The algorithm always checks for a touch, and immediately maps it as starting coordinates X_{new}, Y_{new}. Then a user supposedly moves a finger in a direction he wants the device to go, confirming the end of the swipe (thus assuming this was not a random touch) by double clicking (the final coordinates X_{final}, Y_{final} are saved).

IV. EXPERIMENTAL SETUP

10 people participated (age 23–35) in the experiment. Each person was seated in the wheelchair and asked to do four different control commands using a touch interface (forward, backwards, left, right) presented randomly repeating them 10 times (40 total). A Logitech Wireless Touchpad was used for the evaluation of a touch control within our prototype algorithms. Knowing that some people (especially the disabled) not always can use fingers we decided to test touch interface using different parts of the body (hand finger, nose, leg finger and elbow). Unfortunately the touchpad we used didn't work with the elbow (was unable to sense the touch action due to dryness and roughness of a skin). So these tests were omitted till we'll get a special certified hardware. The position of the touch interface was adapted to the preferences of each person. All users were presented only the touchpad device and had no view of GUI whatsoever (available to a supervisor only for debug and monitoring purposes).

V. EVALUATION

Since none of the participants were really disabled and therefore not used to alternative control schemes, most problems were noticed trying to do the experiments with foot's finger, especially for swiping forwards and backwards (translating to commands "forward" and "backward") as it was hard not to press the device itself or keep the leg just above the surface to do the clicking actions. Another problem was the "heat loss", as the capacitive touchpad performed much more poorly with the cold body parts (feet loose heat rapidly) thought this was a hardware limitation and other implementations of touch sensors should not be limited that much. However this time we had to register longer time-frames due to miss recognitions.

More problems were noted while using nose for inputting a command. Most participants had problems doing a double click and afterward swipes. Some reliability problems were noted with oily skin types. We had to clean the surface with alcohol after a 10 swipes for person with oily skin. Practical limitations of this implementation were noted as well: the

device obscured the visibility field.

The most fluent and problem less were the performance of hand fingers, though this was natural as both the hardware and the users were "adapted" to this.

The experimental results of the first algorithm are offered in Fig. 3.

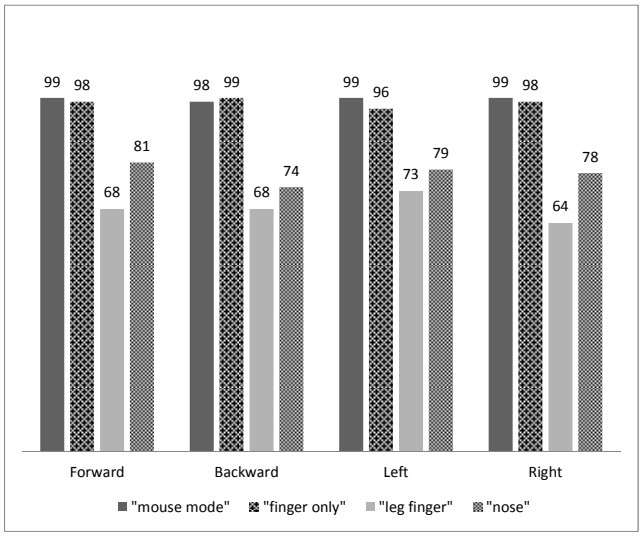

Fig. 3. Average recognition accuracy when using the first touch recognition algorithm.

The reference shown in the first column was done mimicking mouse and using hardware buttons for click actions. All participant were fluent in this mode and no mistakes were noted (~99 % accuracy), as this scheme was very familiar to all of them (typical laptop control scheme). Finger only mode was only ~1.5 % less accurate due to some misses of a double click action although this can be remedied by using a different double click speed for each user. The performance with a foot finger was much lower, on average about 68 %. This again was due to normal users not experienced to this type of action. We expect that a handicapped user, using a leg everyday would be much more accurate as his leg muscles would be much more adapted for precise movements and control. The nose input was recognized at about 78 %. The most problematic issue as mention was the oily nose skin and resulting miss-registers of double click, though this might be solved with other types of touch hardware with other sensor implementations.

The experiment was repeated for a second algorithm (see Fig. 4). The second type of touch action recognition allowed better recognition accuracy for a more severe of input cases. The foot finger actions were recognized 6 % better, the nose actions – 6.5 % better. This can be explained to different working principle. First the user must tap a surface (which is much more easier than double click), the he must swipe (again as many times as he want, as the coordinates for further calculations are used only from the initial tap) and then double click to confirm (also as many times as necessary, the final coordinates are used from the final double click). The performance of finger only mode was similar (a difference if any can be explained to the success of double click, i.e. the favourable time setting).

The comparison of performance (in seconds) is offered in Fig. 5.

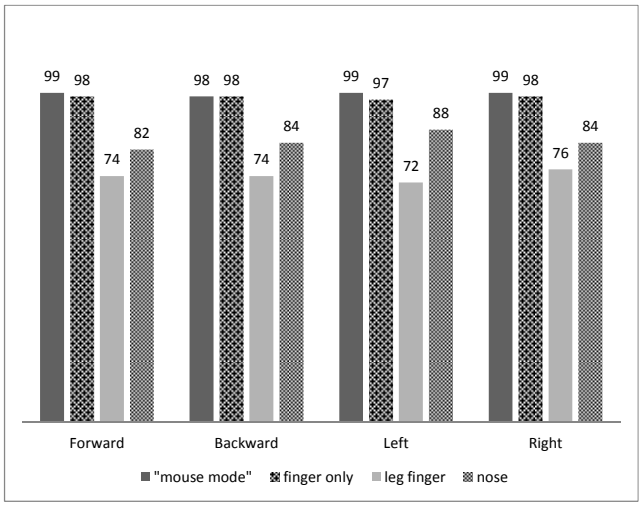

Fig. 4. Average recognition accuracy when using the second touch recognition algorithm.

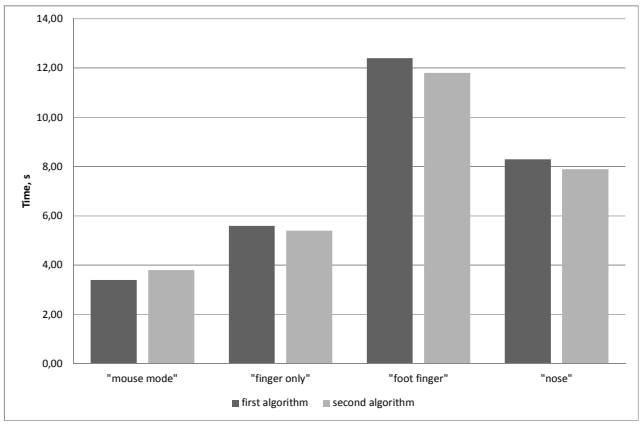

Fig. 5. Average time used for command input.

In most cases the first algorithm was faster than the second (due to fewer actions required). Naturally the fastest mode was the traditional mouse mode (3.6 s.) explainable by the familiarity of this control scheme. Finger only mode was less than 1.9 seconds slower (5.5 s). The alternative inputs (leg and nose) were overall slower to perform (12.1 s. vs 8.1 s.) resulting in slower performance.

After a conclusion of each test all user were asked to subjectively rate the system to their likeness (from 0 (very bad) to 10 (superb)). The results of their evaluation are offered in Table I.

TABLE I. THE RESULTS OF SUBJECTIVE EVALUATION.

	Finger only	**Nose**	**Foot finger**
First algorithm	9.2	6.1	2.3
Second algorithm	9,3	6,4	3.2
TOTAL	9.25	6.25	2.75

As the user were not disabled the results of mouse mode were omitted and even the rest should be quoted as accountant only for the healthy persons (naturally a person with no hands would not rate finger mode better than leg or nose mode).

Overall the users rated finger mode as best as expected (~9.3/10). More notably most somewhat preferred the second algorithm for alternative input modes, probably for the lesser importance of double click quality.

VI. CONCLUSIONS

Two different eyes-free control algorithms were developed to prevent the accidental touch effect. Since none of the participants were really disabled and therefore not used to alternative control schemes, most problems were noticed trying to do the experiments with foot's finger as it was hard not to press the device itself or keep the leg just above the surface to do the clicking actions and while using a nose as some reliability problems were noted with oily skin types.

Finger only mode was recognized ~97.8% accurately (errors were due to some misses of a double click action although this can be remedied by using a different double click speed). The performance with a foot finger was much lower, on average about 71 % due to normal users not experienced to this type of action, while a handicapped user would be much more adapted for precise feet movements and control. The nose input was recognized at about 81.3 % and could be further improved by using other type of touch sensor (not capacitive, thus less sensitive to skin oil effects).

Speaking of performance the first algorithm was faster than the second one, as fewer actions were required to express the wanted action. Finger only mode was executed on average at 5.5 s. The alternative inputs (leg and nose) were naturally slower 12.1 s. vs 8.1 s.).

In the subjective evaluation most users rated finger mode as best (9.25/10) and somewhat more preferred the second algorithm for alternative input modes most likely for the lesser importance of double click accuracy.

REFERENCES

[1] A. A. Bielskis, et al., "Multi-Agent Based E-Social Care Support System for Inhabitancies of a Smart Eco-Social Apartment", *Elektronika ir Elektrotechnika (Electronics and Electrical Engineering*), no. 1, pp. 11–15, 2010.

[2] R. Maskeliunas, "Modeling aspects of multimodal Lithuanian human - machine interface", *Multimodal Signal: Cognitive and Algorithmic Issues,* pp. 75–82, 2008.

[3] K. Koyanagi, T. Oshima, "Force display touchpad: Prototype and pre-test", in *Proc. of the IECON 2011,* 2011, pp. 141–146.

[4] Jung-Hoon Hwang, R. C. Arkin, Dong-Soo Kwon, "Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control", *Intelligent Robots and Systems,* vol. 2, pp. 1444–1449, 2003.

[5] J. F. Kamp, F. Poirier, P. Doignon, "Control of In-vehicle Systems by Gestures", *Gesture-Based Communication in Human-Computer Interaction,* vol. 1739/1999, pp. 159–162, 1999. [Online]. Available: http://dx.doi.org/10.1007/3-540-46616-9_15

[6] A. David Hein, et al., "Touch pad sensor for motor vehicle", Patent number: 7136051, Filing date: 29 Mar 2004, Issue date: 14 Nov 2006.

[7] R. Vilimek, A. Zimmer, "Development and Evaluation of a Multimodal Touchpad for Advanced In-Vehicle Systems", *Engineering Psychology and Cognitive Ergonomics,* vol. 4562/2007, pp. 842–851, 2007.

[8] A. Hiscox, B. Hallowell, J. D. Enderle, "Tap-Tap environmental controls unit", in *Proc. of the Bioengineering Conference,* 2000, pp. 149–150. [Online]. Available: http://dx.doi.org/10.1109/NEBC.2000.842423

[9] Y. S. Choi, et al., "Laser pointers and a touch screen: Intuitive Interfaces for Autonomous Mobile Manipulation for the Motor Impaired", in *Proc. of the ACCESS Conference on Computers and Accessibility,* ACM Press, 2008, pp. 225–232.

[10] *Synaptics TouchPad Interfacing Guide,* 2001, p. 91.