Evaluation of User Gestures in Multi-touch Interaction: a Case Study in Pair-programming

Alessandro Soro[1,2] asoro@crs4.it

Samuel A. Iacolina[2] samueliacolina@gmail.com

Riccardo Scateni[2] riccardo@unica.it

Selene Uras[1] selene@crs4.it

(1) CRS4 - Center for Advanced Studies, Research and Development in Sardinia - Italy

(2) University of Cagliari, Dept. of Mathematics and Computer Science - Italy

ABSTRACT

Natural User Interfaces are often described as familiar, evocative and intuitive, predictable, based on common skills. Though unquestionable in principle, such definitions don't provide the designer with effective means to design a natural interface or evaluate a design choice vs another. Two main issues in particular are open: (i) how do we evaluate a natural interface, is there a way to measure 'naturalness'; (ii) do natural user interfaces provide a concrete advantage in terms of efficiency, with respect to more traditional interface paradigms? In this paper we discuss and compare observations of user behavior in the task of pair programming, performed at a traditional desktop versus a multi-touch table. We show how the adoption of a multi-touch user interface fosters a significant, observable and measurable, increase of nonverbal communication in general and of gestures in particular, that in turn appears related to the overall performance of the users in the task of algorithm understanding and debugging.

Categories and Subject Descriptors

H.1.2 [**MODELS AND PRINCIPLES**]: User/Machine Systems – *Human factors*.

General Terms

Human Factors, Verification.

Keywords

Evaluation Methods, Multi-User Interaction / Cooperation, Multi-touch, Pair Programming, Gesture.

1. INTRODUCTION

Multi-touch interaction has been a topic of research since the mid-eighties [7][10][11], but it's with the recent work of Han [5] than this interaction paradigm has become popular and multi-touch interaction is now so often taken as an example of natural interface.

However, applications based on this interaction paradigm are still in a phase of creative envisioning (see for example [21][19]) and little, if any, study exists on the real advantages of direct manipulation in traditional application fields; For example, Owen and colleagues [15] explores the advantage of bimanual input on a curve matching task; Patten and Ishii [16] presents a study that compares the strategies (and effectiveness) of spatial organization with tangible and traditional user interfaces.

These studies let foresee an advantage of direct manipulation, and by extension of multi-touch tables, over traditional desktop for very specific tasks that have in common a certain physicality, but don't settle the point on whether or not surface computing can replace the desktop in traditional work or learning scenarios.

A generally accepted, while hard to quantify, advantage of multi-touch tables and walls over desktops is their being inherently multiuser: people cooperate to the task at hand, sharing or negotiating the use of the device in a natural manner. Depending on the specific task it is easy to observe an increase of nonverbal communication (gestures, body postures, facial expressions, etc.) that, although not intuitive at all, are proven to have a positive impact on many cognitive processes (see below, related work).

Gestures in particular represent an easy to measure virtuous practice that in desktop computing appear limited almost exclusively to pointing with hand or finger, while observing users of multi-touch tables it often happens to see fluent, dual-handed metaphorical gestures.

This raises the questions we tried to answer. Is there any practical advantage (e.g., in terms of efficient problem solving) when using a natural interface? More precisely: is multi-touch better than the desktop for some traditional application? Moreover: can gesticulation be used as a suitable signal of natural interaction justifying the chain that more gestures provoke a more natural, and, thus, better interaction?

We opted to experiment with pair-programming [18]. It is a practice strongly recommended by agile methodologies and, thus, represents a realistic and not artificial test-bed both for desktop and multi-touch setting. Additionally, gesticulation, which we aim to observe, is more easily, though not exclusively, triggered during group-work.

We will be able to show how interacting in a multi-touch environment determines quantitative increase of nonverbal communication in general and, especially, of gestures, that in turn appears related to the overall performance of the users in the task of algorithm understanding and debugging.

The rest of the paper is organized as follows. Section 2 discusses in depth related work, with emphasis on gestures in human computer interaction and in related disciplines, such as psychology and anthropology. Section 3 describes the experimental setting.

ICMI'11, November 14–18, 2011, Alicante, Spain.

Section 4 presents the results and discusses their implications. Finally, in section 5 conclusion and further work are presented.

2. RELATED WORK

Gesture in Human Computer Interaction has been variously addressed by computer scientists, initially as part of multi-modal user interfaces (following seminal work by Bolt, [1]), and more recently, as a dominant aspect of, to name a few themes, tangible interaction [6] kinetic interaction [2], emotions recognition [3].

However, research in gestural interaction so far has concentrated rather on sensing gestures, (e.g., Pavlovich and colleagues [17]; Wu and Huang [20]) than on defining what a gesture is and how to give meaning to gestures in interactive systems often tailoring this issue to the specific needs of applications (e.g., pen computing) or technologies (e.g., multi-touch displays). Additionally, gesture has been mostly regarded to as an alternative, rather than a companion, to other input devices, that allows a more *natural* form of interaction.

On the other hand, gestures and non-verbal communication, together with those human activities and social interactions which it is functional to, has been the subject of deeper investigation by anthropologists and psychologists.

Among the many movements that we perform with our body, gestures are, according to Kendon, intentional excursions that are meant to convey some message [8] and can be classified along a continuum that spans from gesticulation to sign languages, passing through speech linked gestures, emblems and pantomimes. McNeill underlines how such categories are in relation with the presence/absence of speech and linguistic properties; gesticulation is (almost) always accompanied by speech, emblems may or may not, while sign languages (almost) never are. Additionally gesticulation doesn’t have any linguistic property, emblems have some linguistic properties, sign languages have full linguistic properties [13].

This cognitive role of gestures has an analogous in manipulations. Manipulations are actions performed on objects in order to change the state of the world. Their potential is being explored in the context of tangible interaction, and the role that manipulations play in human cognition has been explained by Kirsh and Maglio [9] in terms of epistemic and pragmatic action, the former being actions performed in order to improve cognition, where the latter are planned and performed to reach a specific goal.

The advantages of such behavior are that: (i) the complexity of the task is moved from the head of the user to the world, available strategies and possible solutions to a given problem appear at a glance; (ii) the (limited) resources of attention and memory are not wasted to concentrate on the strategy and can be used to explore alternative solutions; (iii) such exploration performed by means of manipulations on the world (or tools) are easier (i.e., they require less cognitive effort) and faster (i.e., they require less time) than it is to do the same mentally.

There is also strong evidence that epistemic action increases with skill [12]. This means that in HCI this is not a behavior of naïve computer users, but rather a powerful feature to leverage in interaction design. Tangible and multi-touch interaction, both build on this concept of providing direct manipulation of physical/graphical objects.

But interfaces don't define the interaction. Users' behavior is far more rich of what can be recognized and supported by the interface. In [4] Goldin-Meadow and co-workers show that gesturing lightens cognitive load while a person is thinking and explaining how she solved a math problem, resulting in an improved performance in a short term memory exercise. Similarly, Cook and co-workers [22] show how gesturing during a learning session helps children retain the knowledge gained.

Nonverbal communication in a group is also a positive behavior indicating healthy cooperation. Morrison and colleagues [14] show how the introduction of an electronic patient record in a hospital can disrupt some virtuous practices, partially voiding the benefits of the digital support.

3. EXPERIMENTAL SETTING

A convenience sample of 44 people participated to this study, age 20-35, all students of computer science or ICT professionals, thus quite literate in computer programming.

Working in pairs (see Figure 1), the testers were asked to review 7 snippets of C code (1 demo, and 6 exercises), each one containing a bug, and to point out the bug to an assistant. The review of the code snippets was performed through a very simple interface implemented with the identical look and feel both for the desktop and for the multi-touch environment.

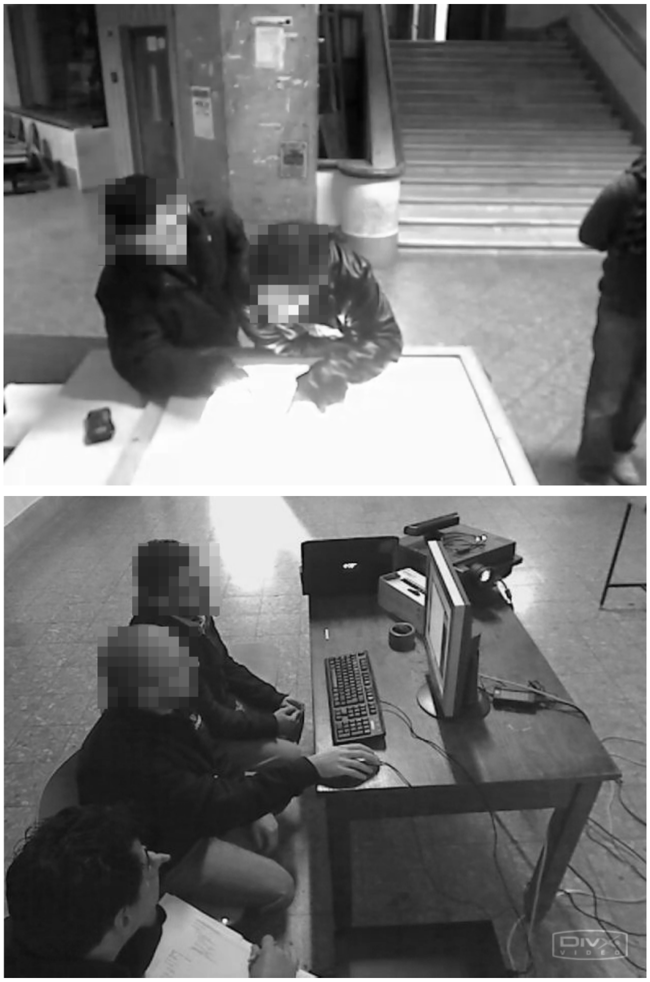


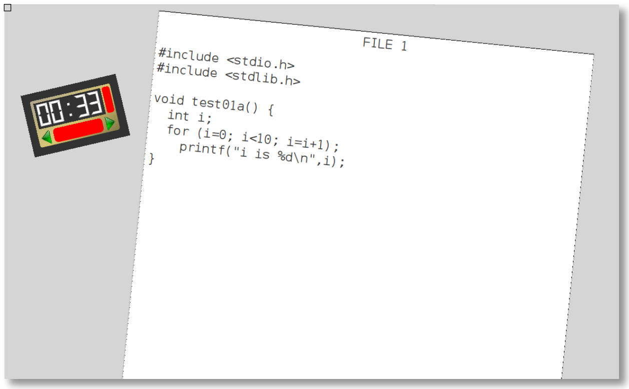
Figure 1: Testers participating at the experiment on the Multi-touch Table (top) and at a traditional Desktop (bottom).

The appearance of the graphical interface is shown in Figure 2; it consists of

- a square text-area that shows one snippet of code at a time. The snippets of code are short enough to fit the visible area, so no scrolling is ever needed and no scrollbars are thus provided;

- a small control panel with a timer, and buttons to jump to the next/previous exercise

Multi-touch functionalities where enabled on the MT table and simulated with keyboard/mouse combinations on the desktop. However, in practice, testers seldom manipulated the interface, except for hitting the 'next' button.



```
#include <stdio.h>
#include <stdlib.h>

void test01a() {
  int i;
  for (i=0; i<10; i=i+1);
    printf("i is %d\n",i);
}
```

Figure 2: The Graphical User Interface of the program used for the tests, both at the Multi-touch and at the Desktop.

3.1 Pre-test briefing

Before the beginning of a test session the testers where briefed on the purpose and method of the research. We had great care to specify that the goal of the work was to evaluate the quality of the tool (Desktop vs Multi-touch) and not the ability of the users. The need of a video recording was justified by explaining our need to monitor 'collaboration and non-verbal communication' but without explicitly mentioning *gestures*, or their supposed connection with efficient problem solving. The testers where then encouraged to cooperate to the solution of the problems.

The testers were also informed that:

1. every snippet contains one (and only one) bug;
2. the bug is not in the syntax, but in the logic of the code;
3. comments (where provided) are not misleading;
4. bugs, although trivial to explain, where sometimes well concealed, and intended to be difficult to spot;
5. finally, although no time constraints were given, the testers were informed that the whole test required between 15 and 25 minutes on average. This was not intended to fix a goal for the performance, but to prepare the testers to the effort needed to complete the test.

Of course all participants were given written warranty of privacy and non-disclosure of videos and disaggregated data.

3.2 Test session

The 44 testers (spontaneously organized in 22 couples) were then asked to complete the experiment. 11 tests were run at the Desktop and 11 were run at the Multi-touch table, the assignment to one or the other setting was performed randomly. The F-test was used to verify if a significant difference exists between the two methods, multitouch and desktop. Note that the same 7 exercises were administered at the 2 settings.

Of the 7 snippets of C code, the first one was intended as a *demonstration* to get into confidence with the interface and clarify latest doubts; results are not taken into account in the following discussion.

For each one of the remaining 6 snippets, the testers had to perform the following:

1. examine the snippet for as much time as needed, discussing, if necessary, to decide what the bug was;
2. as soon as an agreement was reached on the exercise, press a pushbutton (that turns green) on the control panel;
3. testers could then point out the bug to an assistant, who annotated it in a block notes, without either confirming or refusing the answer;
4. by pressing a pushbutton on the control panel the testers could then proceed to the following exercise.

Note that in both settings:

- the interface didn't allow any editing of the C code; so the users were not able to correct the error;
- since the assistant did not comment on the proposed solution, the test actually measures the time spent before reaching an agreement, we did not measure the accuracy (i.e. if the testers positively solve the exercise or not) of the exercise; thus, wherever in the rest of the paper we talk of *solving* an exercise it should be clear that we mean *reaching an agreement on the solution*;
- the cases in which the testers were not able to reach an agreement (either on the correct or on a wrong answer), were also included; in a sense this results indicate the time spent before deciding that additional tools/information was needed to positively solve the exercise; of course such cases should better be taken into account in a deeper investigation (see later, Conclusion and Further work)
- the testers hit a button after reaching an agreement and another one to switch to next exercise, thus the time spent in reporting the bug to the assistant is known and has been expunged in the following discussion.

3.3 The 6 code snippets

The various exercises have been designed to be of increasing complexity and length (and in general took increasing time to solve). The exercise can be divided in 4 categories, and were administered in the same order in which they are described below:

Type 1: controversial exercises such as the one below are likely to cause debate between the testers.

```
void test2() {
  int i;
  for (i=0; i<10; i=i+1)
    if (i=2)
      printf("i is 2\n");
    else
      printf("i is not 2\n");
}
```

In the specific case the use of an assignment as argument of a truth evaluation, though not syntactically wrong, is typically deprecated. There are exceptions however, and the testers spent time discussing whether or not the use of such construct was acceptable in the context of the exercise.

Type 2: slips or careless errors are very common in everyday programming and are easily spotted since often result in meaningless or inconsistent code.

```
void test3() {
  int i;
  i = 0;
  while (i < 10);
    i = i + 1;
  printf("Finished. i = %d\n",i);
}
```

In this case the body of the `while` construct is actually an empty statement (because of the semicolon), resulting in an infinite loop.

Type 3: pattern matching error are those ones that require visual memory or recognition, and represent a class of errors almost unknown to modern programmers, thanks to the use of visual editors that provide syntax highlighting. Examples include misplaced parentheses due to a wrong indentation, and comments opened and not closed, such as the example below.

```
 1 void test4() {
 2   int i;
 3   for (i=0; i<10; i=i+1)
 4     /* check the value of i */
 5     switch(i){
 6       /* is i 0? */
 7       case 0: printf("i is 0\n");
 8               break;
 9       /* is i 1?
10       case 1: printf("i is 1\n");
11               break;
12       /* now the default case */
13       default: printf("i is more than 1\n");
14   }
15 }
```

Most modern editors would help the programmer to find the error here: the comment in line 9 is not closed at the end of the line, and runs through to line 12, voiding in practice the body of the function. Without the help of syntax highlighting, the testers were forced to check the syntax of comments, which is trivial in practice, but not intuitive.

Type 4: algorithm understanding exercises are those ones for which the most effort were required. The bugs presented were in practice overrun of array indexes, such as in the example below.

```
void bubble_sort(int array[], int n) {
  int i, j;
  // sort array of length n
  for (i = (n - 1); i > 1; i++) {
    for (j = 0; j < i; j++) {
      if (array[j] > array[j + 1]) {
        // swap values
        int tmp = array[j];
        array[j] = array[j + 1];
        array[j + 1] = tmp;
      }
    }
  }
}
```

Here the outer `for` cycle will never end and causes an array overrun on the subsequent instructions. Testers were able to solve the exercise only after understanding (or recollecting from previous study) the basic logic of the algorithm.

3.4 Data collection

Data collected during or following the test are:

1. the time spent on each exercise;
2. the proposed solution, that may or may not be correct;
3. the video footage of the whole session.

These were used in the analysis described in the next section. Other data gathered, but not discussed in detail here are:

1. whether or not the testers were able to reach an agreement on the solution of the exercise;
2. subjective scores of the difficulty of each exercise.

This information, as we already noticed, will be subject to further investigation on the accuracy of the performance and on the subjective perceived difficulty of the exercises in the two settings.

3.4.1 Analysis of the video logs

To better understand the role of gestures in collaborative work we have analyzed the video logs of the test sessions in order to count the gestural events. As pointed out in Section 2 there is strong evidence that a fluent gesticulation has a positive influence, among others, on short term memory [4] and learning [22].

We claim that a similar behavior exists in solving complex tasks such as the one considered here, and that a system that allows (or encourages) a fluent gesticulation allows better performances.

The video collected were annotated using Anvil [23], a platform for multilayered annotation of video with gesture, posture, and discourse information.

3.5 Experimental Hypotheses

As mentioned earlier, our research questions are two:

1. Is there any practical advantage (e.g., in terms of efficient problem solving) when using a natural interface? More precisely: is multi-touch better than the desktop for some traditional application?
2. Can gesticulation be used as a suitable signal of natural interaction (i.e., the more gestures, the more natural, and the better interaction)?

Hence the null hypothesis relates to question 1.

H1. Participants will be no faster in solving an exercise containing a *controversial* bug, when using the Multi-touch table or the Desktop.

H2. Participants will be no faster in solving an exercise containing a careless error, when using the Multi-touch table or the Desktop.

H3. Participants will be no faster in solving an exercise requiring a *pattern matching*, when using the Multi-touch table or the Desktop

H4. Participants will be no faster in solving an exercise that require *algorithm understanding*, when using the Multi-touch table or the Desktop.

In order to positively answer question 2 we should first prove that the observed difference in fluency of gestures couldn’t be otherwise explained:

H5. Participants will gesture with no more of less fluency (measured as gestural units/time) at the Desktop or at the Multi-touch table.

Further hypotheses, showing if fluency of gestures has any direct impact on efficient problem solving (i.e., couples with more fluent gesture actually perform better), or a deeper exploration in the nature of gestures involved in this specific task (e.g., what *pantomimes*, *icons, metaphors* were used in addition to *deictics* that helped the participants who scored the better results) are outside the scope of this paper.

4. RESULTS AND DISCUSSION

The experiments show that people perform significantly better at the multi-touch table (for the task examined) than at the desktop for some of the exercise, namely those ones involving cooperation, discussion, and, more generally, exchange of communicational information.

Do participants perform better when solving an exercise containing a *controversial* bug, when using the Multi-touch table than the Desktop? As shown in Figure 3 tester scored slightly better performances at the Multi-touch; the difference is significant, F(10,10) = 4.72. Hypothesis ***H1*** should then be rejected.

The analysis of results of exercise does not show any significant difference between the Desktop and the Multi-touch, F(10,10) = 1.46, n.s. Figure 4 shows means and standard errors for the results of the experiments.

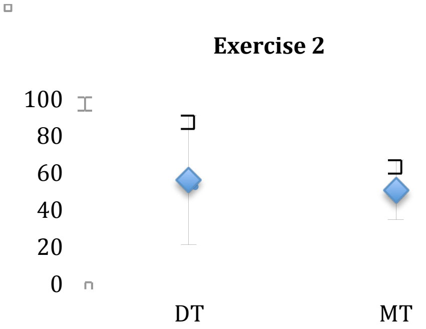


Figure 3: Time (in seconds) to solve exercise 2 (controversial) at the Desktop (DT) and at the Multi-touch (MT). Error bars indicate standard error.

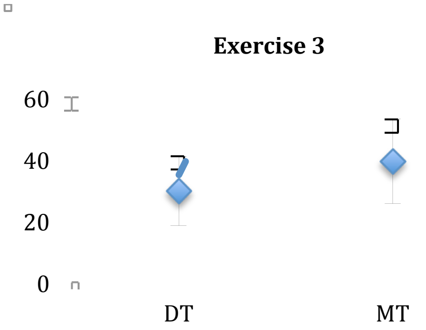


Figure 4: Time spent to solve exercise 3 (slip error).

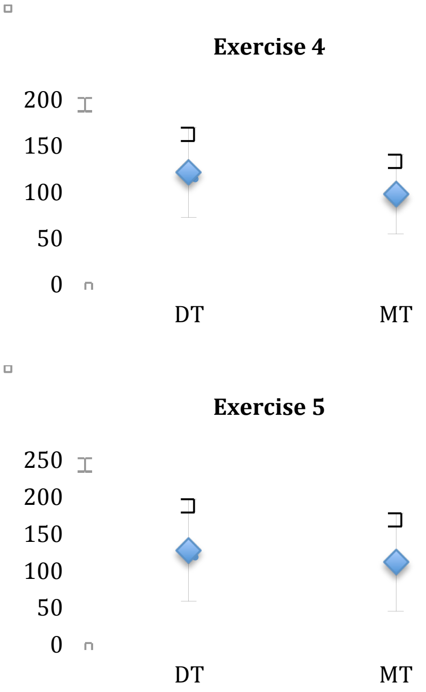


Figure 5: Time spent to solve exercise 4 and 5 (pattern matching exercise).

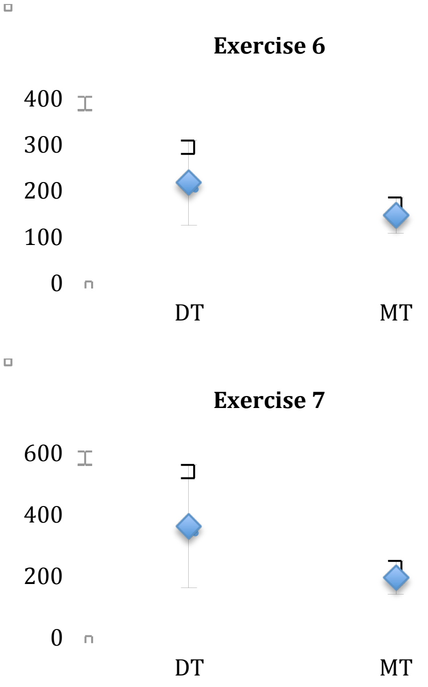


Figure 6: Time spent to solve exercise 6 and 7 (algorithm understanding).

Similarly, no significant difference was observed in the execution of exercises 4 and 5, both containing errors requiring a *pattern matching*: precisely: F(10,10) = 1.29, n.s. for exercise 4 and F(10,10) = 1.08 for exercise 5. Figure 5 shows the results.

Finally, exercise 6 and 7 required the most effort from the testers (as shown by the longer time to solve on average, Figure 6), and the Multi-touch setting allowed a tighter cooperation resulting in a significant better performance: F(10, 10) = 5.56 for exercise 6, F(10,10) = 13.50 for exercise 7.

The timing are summarized in Table 1.

Table 1: Time spent on average on each exercise. Desktop (left column) and Multi-touch (right column) performances are compared.

Exercise #	Avg. Time (Desktop)	Avg Time (Multi-touch)
2	55.18	49.91
3	29.64	39.18
4	119.27	95.82
5	125.64	109.36
6	213.55	143.09
7	356.55	190.80

4.1 Gesture Fluency

Our last test was aimed at showing if users show a difference behavior with respect to gesture fluency in the MT and DT settings. We observed proper gestures according to the related literature given in Section 2. In particular:

- Only movements of the hands were counted as gestures, thus excluding nodding and changes in body postures; specifically, *pointing with the mouse was not counted as gesturing*; in fact, mouse pointing is not a proper gesture and comparison to previous work is problematic. Additionally, we can't assume the visibility of the mouse gesture to the other user, i.e. there is no clear communicative intent (see later).

- Movements of the hands were counted as gestures when they had a clear communicative intent: folding the hands together is not a gesture; pointing, mimicking an action, and counting with fingers are all considered gestures;
- Gesture phrases were counted as their atomic components where possible; for instance, when a tester points a section of code, then another to show correlation, and finally makes sharp movements to show progress, even if these three movements are executed without any visible pause, were counted as 3 separate gestures.

We, thus, introduced for simplicity a measure of *gesture fluency*, as the number of gestural events per second of both testers, and, for each one of the 22 couples, we counted the gesture events of both testers. The gesture fluency of a couple is the total number of gestures performed divided by the total time spent solving the 7 exercises. Results are shown in figure 7.

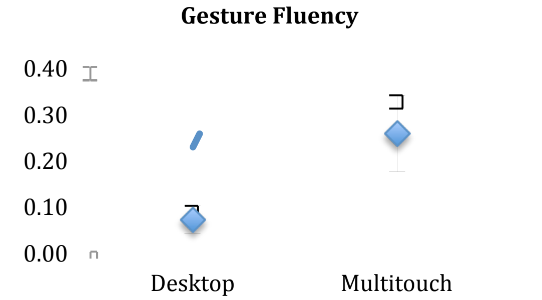


Figure 7: Measures of gesture fluency in the two interaction environments.

The experiment shows that participants use significantly more gestures when using the Multitouch than when interacting at the Desktop, F(10,10)=7.70; thus we can reject hypothesis **H5**.

4.2 Discussion

The results shown above indicate that while working at the multitouch table people perform better than at a traditional desktop, and such improvement is associated to an increased gesticulation.

Some remarks are due here. Not all types of exercise seem to benefit from the adoption of a multi-touch system, in particular snippets containing *careless errors* (exercise 3) and *pattern matching* (exercises 4 and 5) were not significantly affected by the different setting. *Controversial* exercises (such as exercise 2) are better addressed at the multitouch, where a tighter cooperation is possible. This is hardly a surprise, since this sort of problems requires discussion and sometimes negotiation between the users.

The results obtained for exercises 6 and 7 (*algorithm understanding*) are perhaps less intuitive and their implications in the design of interactive applications deserve some attention.

On one hand this work gives a further confirmation of the already observed connection between gesturing and problem solving. In this case an improvement in the interactive systems did not involve improvements *in the interface*, but rather the design of a work-setting more suitable for cooperation, and fluent gesturing was taken as a metric for the cooperation itself.

On the other hand, one can notice that the exercises taking the most benefit from the multitouch setting were the more difficult among the 7 administered, and still were trivial with respect to the typical problems that programmers face in daily work. Our experiments suggest that multi-touch tables, encouraging cooperation, help people express their potential, thus resulting in a better performance. The registered difference in performances for code understanding and debugging time could make a significant difference in many practical cases. If confirmed, these results may help reconsidering the design of our offices and programming labs towards a more widespread adoption of tabletops, that today are mostly regarded as research prototypes and curiosities.

Some further empirical observations are worth mentioning here. Our metric of *gesture fluency* was suitable for the work at hand, but hides the real complexity of gesture phrases. If the gesture largely more exploited by all participants was *pointing* with one finger, others where frequently observed:

- Gestures indicating progress or continuity, both single and dual-handed, are executed moving the hand(s) on a circle or sharply from left to right; such gestures are not easily performed when sitting, and not surprisingly they are less frequently seen at the Desktop;
- Some gestures are performed primarily for communicating, they are a sort of *visible words*; as such they have to be performed in a well-defined and visible space; again, such space (close to the screen) is easier to reach at the Multitouch than it is at the Desktop;
- At the Multitouch *pointing* with the finger was sometimes used to negotiate the attention of the mate; testers often pointed at the same point on screen as to reinforce and confirm a gesture; this behavior was not observed at the Desktop;
- In one case a tester asked if she could use paper and a pencil, which was not possible, actually; several participants at the Multitouch setting were observed while mimicking the use of paper and pencil on the palm of the open hand.

5. CONCLUSION AND FUTURE WORK

We have shown how the adoption of a multi-touch user interface can lead to a significant, observable and measurable, increase of nonverbal communication in general and of gestures in particular, that in turn appears related to the overall performance of the users in the task of code understanding and debugging. Our results indicate that working at the multi-touch table people perform better than at a traditional desktop, and such improvement is associated to an increased gesticulation. Users at the multi-touch outperformed desktop user for specific classes of problem, and such gap corresponded to 4 as much gestural events, indicating an improved cooperation.

As noted throughout the work however, several questions remain open.

- We didn't observe the accuracy of the solutions proposed. For the scope of this research a problem was considered solved when an agreement on the proposed solution was reached; though in principle cooperation and discussion lead to accurate results, a precise measure of such accuracy is likely to expose new insights;
- The choice of gestures to observe was arbitrary, though shared in literature; for example, pointing with the mouse is a common behavior at the Desktop, whose impact should be evaluated;
- How strong an interrelation exists between gestures and efficiency/accuracy? Do couples that show more gesture fluency perform better?

- What new insights may come from a more detailed analysis of gestures; gesture fluency doesn't capture the richness of expression that emerges at the multitouch table, where dual-handed symbolic gestures are often used compared to bare single-handed deictic that form the majority of gestures at the Desktop.

In future work, we wish to further explore the relationships between natural interfaces (including, but not limited to multi-touch tables) and user efficiency in traditional tasks, with the aim of understanding the applicability of new interface paradigms to real workplace and learning environments. In particular, the relationship between gesture fluency and performance at the single exercise is likely to throw light on how people address the task considered (algorithm understanding and debugging).

Our ultimate goal is to identify new practices and guidelines for designers and practitioners, to help evaluate new natural interfaces, and testing design choices respect how they encourage or obstruct those virtuous practices such as gesturing and non-verbal communication.

6. ACKNOWLEDGEMENTS

Some of the C code snippets were adapted from fragments available online under GNU GPL or analogous licenses. Snippets 1-4 were adapted from [24].

7. REFERENCES

[1] Bolt R. A.: "Put-that-there": Voice and gesture at the graphics interface . In Proc. Of SIGGRAPH '80: the 7th annual conference on Computer graphics and interactive techniques

[2] Bruegger, P. and Hirsbrunner, B. Kinetic User Interface: Interaction through Motion for Pervasive Computing Systems, in Proc. of the 5th International on ConferenceUniversal Access in Human-Computer Interaction. UAHCI '09

[3] Castellano, G. and Kessous, L. and Caridakis, G. Emotion Recognition through Multiple Modalities: Face, Body Gesture, Speech. in: Affect and Emotion in Human-Computer Interaction: From Theory to Applications (2008)

[4] Goldin-Meadow, S., Nusbaum, H., Kelly, S.D., & Wagner, S. (2001). Explaining math: Gesturing lightens the load. Psychological Science, 12, 516–522.

[5] Han J. Y.: Low-cost multi-touch sensing through frustrated total internal reflection. In UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology (New York, NY, USA, 2005), ACM, pp. 115–118

[6] Ishii, H., and Ullmer, B. Tangible bits: towards seamless interfaces between people, bits and atoms. In CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems, pages 234–241, New York, NY, USA, 1997. ACM.

[7] Kasday L. R., NJ P.: Touch position sensitive surface, 1981/12/23 1984. 3

[8] Kendon, A. 2004. Gesture: Visible Action as Utterance. Cambridge University Press

[9] Kirsh, D. and Maglio, P. On Distinguishing Epistemic from Pragmatic Action. Cognitive Science 18, 513-549 (1994)

[10] Krueger M. W., Gionfriddo T., Hinrichsen K.: Videoplace—an artificial reality. SIGCHI Bull. 16, 4 (1985), 35– 40. 3

[11] Lee S., Buxton W., Smith K. C.: A multi-touch three dimensional touch-sensitive tablet. SIGCHI Bull. 16, 4 (1985), 21–25. 3

[12] Maglio P. P., Kirsh D.: Epistemic action increases with skill. In In Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society (1996), Erlbaum, pp. 391–396. 3

[13] McNeill, D. Hand and Mind: What Gestures Reveal About Thought. University of Chicago Press, Chicago. 1992.

[14] Morrison, C., Jones, M., Blackwell, A., Vuylsteke, A. (2008) Electronic patient record use during ward rounds: a qualitative study of interaction between medical staff, Critical Care 2008

[15] Owen, R. Gordon Kurtenbach, George Fitzmaurice, Thomas Baudel, and Bill Buxton. 2005. When it gets more difficult, use both hands: exploring bimanual curve manipulation. In Proceedings of Graphics Interface 2005 (GI '05). Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 17-24.

[16] Patten, J and Ishii, H. 2000. A comparison of spatial organization strategies in graphical and tangible user interfaces. In Proceedings of DARE 2000 on Designing augmented reality environments (DARE '00). ACM, New York, NY, USA, 41-50.

[17] Pavlovic V. I., Sharma R., Huang T. S.: Visual interpretation of hand gestures for human-computer interaction: A review. IEEE Trans. Pattern Anal. Mach. Intell. 19, 7 (1997), 677–695.

[18] Williams, L. and Kessle, R. 2002. Pair Programming Illuminated. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[19] Wilson, A. D. and Hrvoje Benko. 2010. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In Proceedings of the 23nd annual ACM symposium on User interface software and technology (UIST '10). ACM, New York, NY, USA, 273-282.

[20] Wu Y., Huang T. S.: Vision-based gesture recognition: A review. In GW '99: Proceedings of the International Gesture Workshop on Gesture-Based Communication in Human-Computer Interaction (London, UK, 1999), Springer-Verlag, pp. 103–115.

[21] Zeleznik, R., Andrew Bragdon, Ferdi Adeputra, and Hsu-Sheng Ko. 2010. Hands-on math: a page-based multi-touch and pen desktop for technical work and problem solving. In Proceedings of the 23nd annual ACM symposium on User interface software and technology (UIST '10). ACM, New York, NY, USA, 17-26.

[22] Cook, S. W., Mitchell, Z., Goldin-Meadow, S. (2008). Gesturing makes learning last. Cognition, 106, 1047-1058.

[23] Kipp, M. (to appear) Multimedia Annotation, Querying and Analysis in ANVIL. In: M. Maybury (ed.) Multimedia Information Extraction, Chapter 19, IEEE Computer Society Press

[24] Love, T. ANSI C for Programmers on UNIX Systems. Cambridge University Engineering Department (available online: *ftp://svr-www.eng.cam.ac.uk/misc/love_**C**.ps.Z*)