To introduce this third experiment, we firstly tested the conflict detection system on the second experiment to see how many users had defined confusing gestures. Then we added this conflict detection algorithm on the initialization phase of our application and made a third experimentation.

5.1 Conflict Detection with Recognition System

A gesture is said to be confusing when the classifier gives it high probabilities of belonging to multiple classes. In other words, a gesture is not confusing when the recognizer gives it a high probability of belonging to a single class, and low probabilities of belonging to other classes. In our application, we state gesture classes as confusing when the third gesture sample (of the initialization gestures) of this class is found confusing by the classifier after learning the two first gesture samples.

A fuzzy inference system, like *Evolve*, is composed of several fuzzy rules that are each associated with a cluster of the input space. If every rule participates in the recognition process of every class, each cluster is mainly associated with a single class. Any gesture that is between two or more cluster, and any clusters that are too close to each other, should be signaled as confusing. Such a confusing gesture will activate several rules to similar levels and will likely be given similar probabilities of belonging to multiple classes.

We use this property to compute a confidence degree as the difference between the activations of the most activated rule ("$rule_{first}$") and the second most activated rule ("$rule_{second}$") normalized by the activation of the most activated rule.

$$confidence = \frac{activation(rule_{first}) - activation(rule_{second})}{activation(rule_{first})} \quad (1)$$

This confidence degree varies between 0, when two rules are equally activates, and 1 when a single rule is activated. It allows us to flag gestures as confusing when confidence is below a defined confidence threshold (0.09 in our case). The choice of the threshold is explained in our paper [17].

When conflicts are detected, we suggest a change of gestures for the corresponding class. If the user doesn't want to change his gestures, then more gesture samples are asked to try to reduce the conflict between classes.

5.2 Utilization of Conflict Detection on Previous Experiment Data

As mentioned above, our first objective is to find out how many users draw potentially confusing gestures in the last experiment. To stay homogeneous, we took only users from group 2 where users were helped by *Customizable Gesture Menus*.

We have developed an application which present all gestures defined by users one by one. Figure 11 shows one specific user of group 2. The histograms present the confidence degree of each gesture. On the top of each histogram, there is a sample of each gesture. By fixing the threshold at 0.05, we detect 2 confusing gestures, and we

can tell that apparently they look very similar, so the conflict detection algorithm works well.

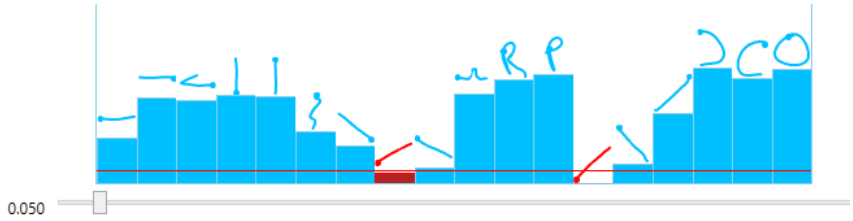

Figure 11. Screen shot of the application showing all 18 gestures of one specific user. The confusing gestures are marked in red while others are in blue since they are all different.

We made this test on all forty users, and then separated them into two categories. In the green category, no user made confusing gestures for the recognizer. In the red category, every user made at least two potentially confusing gestures. Finally we got nine persons in green category and thirty-one persons in red category. This big difference proved that people really need help to avoid making confusing gestures during the definition of their gestures.

5.3 Experimentation 3 with Conflict Detection during Gesture Definition

Our second objective by using the conflict detection is to add it on the initialization phase. Our hypothesis is to see if it really helps on the gestures' definition and if it can bring any advantage on cross-learning results.

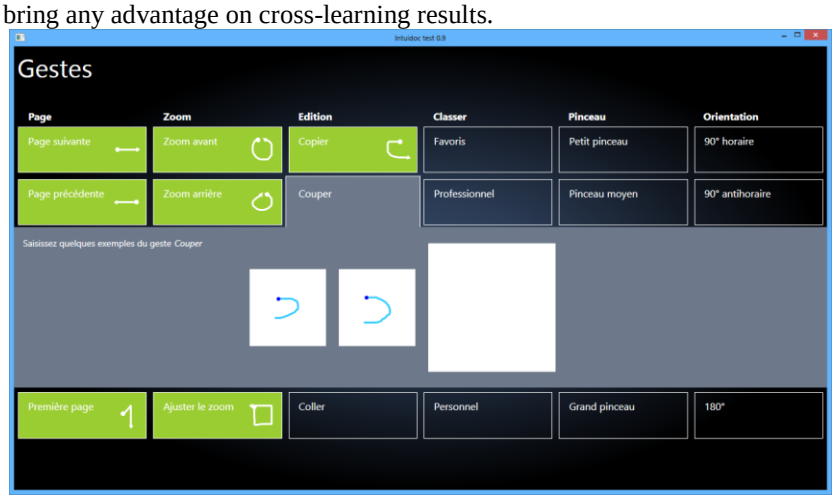

Figure 12. New interface of initialization phase.

The third experimentation is nearly the same as the second one. It followed also the proceedings: initialization phase - test phase 1 - learning phase - test phase 2. We just made some changes on the design of initialization phase. Instead of separating the definition of gestures and practice of gestures for recognition system, we combined them together (cf. Figure 12). By clicking on the label of each command, we reached

the definition box of this command. We needed to give three samples of gesture and one sample would be shown by the side of command's label as a preview. In this way, the user can always have a general idea of gestures that he has already defined, and he can change gestures easily. When there was no conflict detected, all gestures were shown in green. This detection of conflict was checked each time three samples of one command were submitted.

Figure 13. Initialization phase with one detected conflict. In this example, we have the same gesture for the command "first page" (column 1, line 3) and the command "file in favorite folder" (column 4, line 1). These commands are marked in red while other commands already defined are in green because they are different from one another. The user can choose either to enter a fourth sample to one of two confusing commands or to change completely all samples of one command.

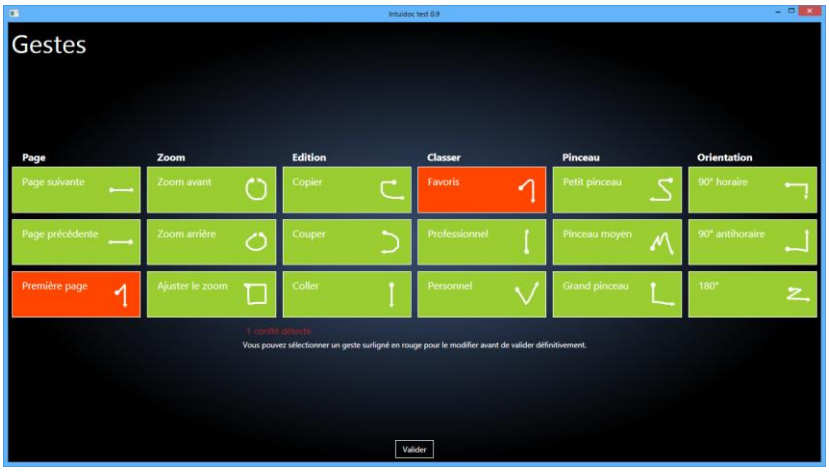

Figure 14. Conflict can also be detected in the end once we click on "validate". If according to him, these gestures are different between them, he can click again on "validate" to force the learning of gestures by the recognition system.

When a conflict was detected, the two concerned gestures would be shown in red as shown in Figure 13. Moreover, a final check would be done once all gestures of commands were defined (cf. Figure 14). If there was any conflict, the user just needed to enter the definition box by clicking on command's label to change the gesture. If from his opinion, the highlighted red gestures were not confusing at all, he could choose to add one sample or more to one command, or to force the recognizer to learn as it was by clicking on "validate".

5.4 Results

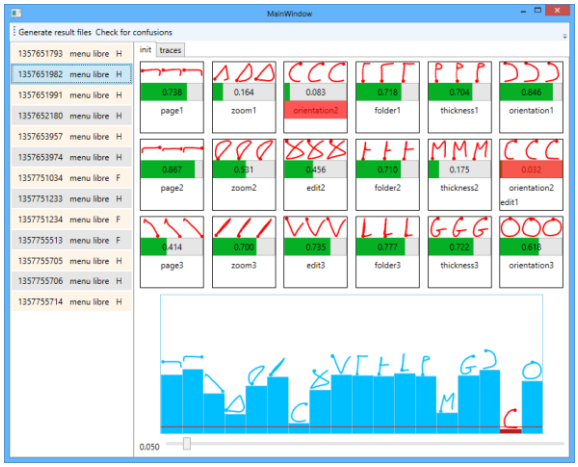

Figure 15. Conflict detected for 2 gestures.

Sixteen users participated in this new experiment. They are around twenty-two-year-old. They are all familiar with computers. In this experiment, users were alerted if they made similar gestures thanks to the conflict detection algorithm. There were always users who decided to force the learning of gestures for recognition system even there was still conflict (cf. Figure 15). But it was up to them to decide so.

We also obtained the learning rates of user (memorization rates) and recognition system (recognition rates). Users from this new experiment are called as Group 2. We compared them to results obtained by group 2 in the second experiment which is named as Group1 here. Group 1 is separated into Group1+ (users who made less than three confusing gestures) and Group1- (users who made at least 3 confusing gestures). We than got twenty seven persons in Group1+ and twelve users in Group1-. We can see that in average score (cf. Figure 16 left part), Group1+ and Group2 are both better than Group1-, either for test phase 1 or for test phase 2. We obtained similar results for mean recognizer learning rates too. But it is not enough to say that our hypothesis is true. As we see in the right part of Figure 16, we have a lot of variation within each group. So we need to make a signification test to justify our hypothesis.

The results of the pairwise tests to compare Group1+ to Group1– and Group2 to Group1– are shown in Table 1 and Table 2. We can see that Group1+ is significantly better than Group1- for almost all measure, while Group2 is significantly better than

Group1- too. We did not obtain any significant difference between Group2 and Group1+. This is as we expected since Group2 was helped by the conflict detection system and it should be as good as Group1+ which made few confusing gestures. From these statistical tests, we can conclude that both Group2 and Group1+ obtained better performance than Group1- in term of cross-learning of the user and the recognizer.

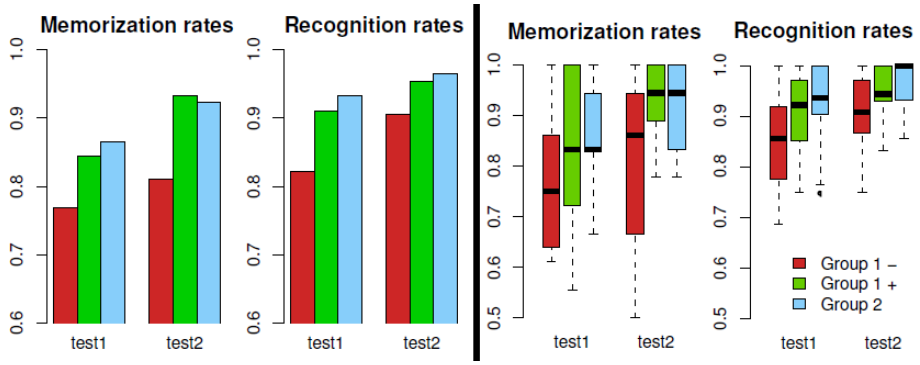

Figure 16. At left, bar plots of users' mean memorization rates and mean recognition rates at two test phases of all groups. At right, Box plots of memorization rates, of recognition rates for all groups.

Table 1. Group1+ compared to Group1-.

Rates	Statistical formulas	Significance
memo-test1	Nothing	Nothing
memo-test2	$\chi^2 = 5.0743$, df = 1, p-value = 0.02428	Group1+ > Group1- Significant
reco-test1	F(1, 13.133)=3.6463, p-value=0.07	Group1+ > Group1- Hardly Sign.
reco-test2	$\chi^2 = 3.8429$, df = 1, p-value = 0.04996	Group1+ > Group1- Significant

Table 2. Group2 compared to Group1-.

Rates	Statistical formulas	Significance
memo-test1	$\chi^2 = 4.0617$, df = 1, p-value = 0.04387	Group2 > Group1- Significant
memo-test2	$\chi^2 = 3.2602$, df = 1, p-value = 0.07098	Group2 > Group1- Hardly Sign.
reco-test1	F(1,15.644)=5.2093, p-value = 0.0368	Group2 > Group1- Significant
reco-test2	$\chi^2 = 4.8478$, df = 1, p-value = 0.02768	Group2 > Group1- Significant

6 Conclusion

In this paper, we analyzed "user & system cross-learning" of gesture commands through several new concepts. We reported three experiments, summarized in Table 3, to support our approach for designing pen-based interfaces for complex application needing more than ten gestural commands: the first concept is to offer users the possibility to personalize their gestures; the second concept is to use *Customizable Ges-*

ture Menus to help users memorize gestures; the third concept is to help users during the definition of their gestures prevent too similar gesture definition. On this last concept, more tests need to be done to consolidate the efficiency of the conflict detection algorithm (because all rates are not significantly better). Moreover the reported results show that using an evolving gesture recognition engine that learns incrementally, starting from few data samples, is a really promising strategy to induce a cross-learning of user and recognition engine. We summarize that personalized gestures can offer an easier way to interact with software on pen-based devices. The handover of software offering the gesture's personalization should be separated into two steps: the help on definition of gestures and the help on memorization during the utilization.

Besides these new concepts, we know that Marking Menus can offer more advantages on interactive devices compared to traditional way with either menu bar or tabular menu, like entering text or parameters [12, 13]. Integration of these extension concepts will be our future work.

Table 3. Overview of 3 experiments.

Exp	Goal	Help on gesture learning	Help on gesture definition	Result
1	Determine if it is important to leave user the freedom to personalize their gestures	Table	None	Personalized gestures are better than pre-defined gestures
2	Determine which form of help suits better user and tabletops to memorize gestures Evaluate the cross-learning of user & recognition engine	Table *vs* Customizable Gesture Menus	None	Menus are equal to table to help users learn gestures Both (the user and the recognition engine) can improve, in the same time, their capabilities of learning
3	Avoid confusing gestures	Customizable Gesture Menus	Conflict detection algorithm	With the help of the conflict detection system on gestures' definition, the user and the recognition engine learn better

References

1. Kurtenbach, G., Buxton, W. The limits of expert performance using hierarchic marking menus. *Proc. INTERACT 1993*, ACM Press (1993), 482–487.
2. Kurtenbach, G., Moran, T.P., Buxton, W., 1994. Contextual animation of gestural commands. Computer Graphics Forum 13 (5), 305–314.
3. Zhao, S., Agrawala, M. and Hinckley, K. Zone and Polygon Menus: Using Relative Position to Increase the Breadth of Multi-stroke Marking Menus. *Proc. CHI 2006*, ACM Press (2006), 1077-1086.

4. Zhao, S. and Balakrishnan, R. Simple vs. Compound Mark Hierarchical Marking Menus. *Proc. UIST 2004*, ACM Press (2004): 33-42.
5. Bau, O., Mackay, W.E. OctoPocus: a dynamic guide for learning gesture-based command sets. *Proc. UIST 2008*, ACM Press (2008), 37–46.
6. Delaye, A. and Anquetil, E. "Hbf49 feature set: A first unified baseline for online symbol recognition" Pattern Recognition, vol. 46, no. 1, pp. 117 – 130, January 2013.
7. Delaye, A., Sekkal, R. and Anquetil, E. Continuous Marking Menus for learning Cursive Pen-based Gestures. *Proc. IUI 2011*, ACM Press (2011), 319-322.
8. Bailly, G., Lecolinet, E. and Nigay, L. Wave Menus: Improving the Novice Mode of Hierarchical Marking Menus. *Proc. INTERACT 2007*. Springer (2007), 475—488.
9. Almaksour, A., Anquetil, E., Quiniou, S. and Cheriet, M. Evolving Fuzzy Classifier: Application to Incremental Learning of Handwritten Gesture recognition Systems. *Proc. ICPR 2010.*
10. Almaksour, A. and Anquetil, E. Improving premise structure in evolving Takagi-Sugeno neuro-fuzzy classifiers. Evolving Systems, 10.1007/s12530-011-9027-0, 2:25-33, 2011.
11. Petzold, C. "Canonical Splines in WPF and Silverlight." http://www.charlespetzold.com/blog/2009/01/Canonical-Splines-in-WPF-and-Silverlight.html
12. Guimbretiére, F. and Winograd, T.: FlowMenu: combining command, text, and data entry, Proceedings of the 13th annual ACM symposium on User interface software and technology, p.213-216, November 06-08, 2000, San Diego, California, United States [doi>10.1145/354401.354778].
13. Igarashi, T., Matsuoka, S., Kawachiya, S., & Tanaka, H. Interactive beautification: a technique for rapid geometric design. *Proc. UIST '97*, ACM Press (1997), 105-114.
14. Norman, D.A., 2010. Natural user interfaces are not natural. ACM Magazine Interactions 17 (3), 6–10.
15. LaViola, J., and Zeleznik, R. "A Practical Approach to Writer-Dependent Symbol Recognition Using a Writer-Independent Recognizer", IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(11):1917-1926, November 2007.
16. Wobbrock, J.O., Wilson, A.D. and Li, Y. (2007). Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes. Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '07). Newport, Rhode Island (October 7-10, 2007). New York: ACM Press, pp. 159-168.
17. Bouillon, M., Li, P., Anquetil, E. and Richard, G. Using Confusion Reject to Improve User and System Cross-Learning of Gesture Commands. *Proc. ICDAR 2013, to be appeared.*
18. Li, P. and Anquetil, E. Graphical Gesture Commands' learning with the help of Customizable Gesture Menus. *Proc. IGS 2013, to be appeared.*
19. Li, P., Delaye, A. and Anquetil, E. Evaluation of Continuous Marking Menus for Learning Cursive Pen-based Commands. *Proc. IGS 2011, 217-220.*