Compensate the Speech Recognition Delays for Accurate Speech-Based Cursor Position Control

Qiang Tong and Ziyun Wang

College of Computer Science and Technology, Hubei Normal University,82, Cihu Road, Huangshi, Hubei Province, 435002, P. R. China
qiangtong99@gmail.com

Abstract. In this paper, we describe a back-compensate mechanism to improve the precision of speech-based cursor control. Using this mechanism we can control the cursor more easily to move to small on-screen targets during continuous direction-based navigation despite the processing delays associated with speech recognition. In comparison, using traditional speech-recognition systems, it is difficult to move the cursor precisely to a desired position because of the processing delays introduced by speech recognition. We also describe an experiment in which we evaluated the two alternative solutions, one using the traditional speech-based cursor control, and the other using the back-compensate mechanism. We present the encouraging evaluation results at the end of this paper and discuss future work.

Keywords: Speech recognition, delays, navigation, mouse, cursor control.

1 Introduction

With the development of practical speech recognition systems and tools such as IBM ViaVoice and Microsoft Speech SDK, a user can operate a computer and use computer applications with only voice without traditional input devices such as keyboards and mice. This is important for individuals with physical disabilities and limited abilities to use keyboards and mice. Speech-recognition enabled computer applications can also help users who need hand-free solutions when their hands are engaged in other tasks.

Computer mouse has been one of the most important computer input devices since the 1960s. Modern computer operating systems, such as Windows, Mac OS and Linux, all provide WIMP (Windows, Icons, Menus and pointing devices) style interfaces. Almost every computer has a mouse and a keyboard as standard input devices.

Combining speech recognition with WIMP style interfaces to create speech-based cursor control is very useful. It can help individuals with physical disabilities to use any existing windows-based applications normally as opposed to only use special applications designed for persons with disabilities.

One crucial problem with speech-based cursor control application is that speech recognition always has delays during it works; the user must complete the utterance of a word and wait for the recognition results. The speech-recognition result is only

J.A. Jacko (Ed.): Human-Computer Interaction, Part II, HCII 2009, LNCS 5611, pp. 752–760, 2009.

available at the end of the utterance, not at the start; this shortcoming limited the use of speech based-applications especially for the continuous direction-based navigation of speech-based cursor control. For example, the user specifies "Move Left" and the cursor begins to move to left, when the cursor gets to the target the user says "Stop" intending to have the cursor stop at the target. In speech-based systems, the cursor often misses the target because the speech-recognition delays effect. The delay effect becomes unacceptable when the user's speech speed is slow ("stoooop"), the cursor will stop farther away from the target.

This paper proposes the use of a compensate mechanism to help speech-based cursor control to remedy against the delay effects with speech-recognition.

2 Related Work

Sears, Lin, and Karimullah [5] provided a detailed analysis of the delays associated with the execution of speech-based commands, and they designed a predictive cursor to help user to estimate where to issue the "Stop" command before the actual cursor got to the target, but it failed to prove beneficial. This is because they hypothesized that the predictive distance for a user is constant. They calibrated the offset used for the predictive cursor for each individual user before they conducted the tests, but a user can't always issue the same commands using the same amount of time. The research, however, found that cursor speed, target size, and speech recognition delays and errors are the most critical factors in achieving precision using speech-based cursor control.

Dai, Goldman, Sear, Lozier [2] presented a grid-based cursor control method. In this grid-based system, the screen was divided into a 3x3 grid numbered one through nine in row-major order. The user spoke aloud the number of the grid that contained the target, and then the chosen grid was recursively divided into a smaller 3x3 grid, and the user continued to speak out the number of the target grid, fine-tuning the target position to move the cursor to reach the proper location eventually. Additional commands were provided to move the whole 3x3 grid in four directions or back up a recursion level if the user made a mistake. This grid-based approach can be efficient in moving the cursor to a point on the screen, but it does not allow the user to move the cursor continuously like in a normal WIMP environment. Further more, this approach works in applications where the cursor is only used for picking targets on the screen; it doesn't help in applications that require the use of the continuous motion of the cursor, such as drawing a line in a painting program.

Igarashi, Hughes [3] showed how nonverbal voice can be used for interaction control, where the user controlled the application directly using continuous voice command, and the system provided immediate feed back. For example, one could say "Cursor up, ahhhhhh…", and the cursor would continually move up while the "ahhh" sound continued, it will stop at once when the "ahhh" sound ended. Subsequent systems have used similar non-verbal voices for continuous input [1][2][6][7], primarily for mouse pointer control. The limitation of these techniques is that it requires an unnatural way of using the human voice.

3 Backward Compensate Cursor Control

We focus on speech-based cursor control for continuous directional navigation. Because of the delay associated with spoken commands, a moving cursor will not stop immediately when the "Stop" command is issued; it will pass the target for some additional distance before it stops, assuming the beginning of the utterance is when the cursor SHOULD stop. Our solution is to make cursor jump back an appropriate distance to compensate the additional distance the cursor travels after the "Stop" command is issued, therefore the cursor will stop at the desired position after the compensation.

Sears, Lin, and Karimullah [5] have mentioned a compensate solution. In that solution, the compensated distance is calculated by determining the average delays from historical usage data. Because true delay for each command never remains a constant, the compensated delay feels artificial and inaccurate.

In fact, the length of command utterance not only varies from person to person, it also varies for the same person from time to time, for example, when a person's speech speed changes due to fatigue or excitement. Therefore the compensate distance should not be a constant value.

In our solution, we detect the delay associated with spoken command such as "Stop" every time when it is issued. Because the speed of the cursor is known, we can calculate the extra distance the cursor travels every time, we then make the cursor jump backwards this additional distance to the correct position when the command is issued. The process is illustrated in Fig.1.

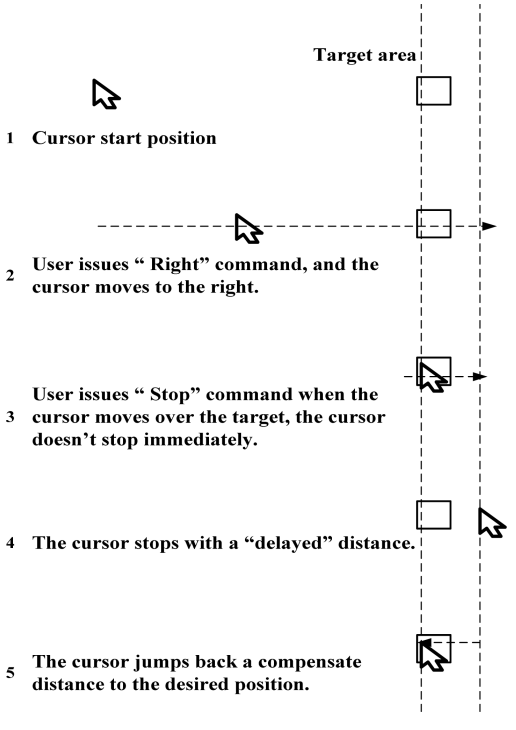

Fig. 1. Illustration of the compensate cursor control

The key contribution of this solution is that the compensate distance is not a constant value, but it is calculated with the actual delay based on when the spoken command is issued and when it is recognized. For example, when the user issues the command STOP as "Stop" (speaking normal speed) and "S-t-o-p" (speaking slowly), the delays are very different, therefore the compensate distances are very different too, the second one is a bigger compensation whereas the first one is smaller.

We implemented the speech-based cursor control with compensate cursor solution as follows:

Our application supports 4 directional voice commands: Left (Move left), Right (Move right), Up (Move up), Down (Move down), and two action commands Stop and Click (mouse left click). Of curse we can add other commands such as Double Click, Drag and other directional commands if necessary.

The cursor moves at a rate of 100 pixels per second.

When the cursor is not moving, if any directional command is issued, the cursor begins to move; when the "Stop" command is issued, the cursor stops and jumps back a compensated distance (as shown in Figure 1); when the cursor is moving, if a different directional command is issued, the cursor does a 3-step adjustment: the first step is to stop, the second step is to jump back a compensate distance, and then the third step is to begin a new direction movement according to the new directional command.

4 Experiment

4.1 Participants

Sixteen HBNU students (8 females and 8 males) volunteered to participate in a usability study. They all speak Chinese and have no hearing, speech, or cognitive impairments. Their average age was 21. They were divided into two groups with one group using a compensate speech-based cursor application and the other group using a normal voice-controlled cursor application.

4.2 Equipment

An IBM ThinkPad running Windows XP was used. The LCD screen had a diagonal size of 14.1 inches and the display resolution was set to 960x600 pixels; Our speech cursor control applications were developed using Delphi 7.0 and Microsoft speech recognition engine 5.1 through Microsoft Speech API. All participants used a headset mounted microphone when testing the application. A set of custom applications were developed using Delphi, presenting 3 different sizes of targets. The applications all automatically recorded the voice command events, and various timing such as selection time used to pick the target.

4.3 Experiment Design

The purpose of this experiment is to determine the benefits of our compensate mechanism in speech-based cursor control systems. The two sets of test applications are only different in whether the delay is processed when stop or change direction command is issued. For accuracy, we designed a script for every test subject to follow in

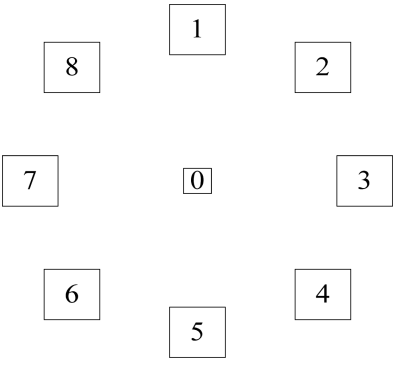

Fig. 2. Target direction relative to the cursor start position

using both applications to complete the same list of tasks in the same order on a screen layout illustrated in Fig. 2:

- use speech command to click the center button 0 to start the task, control the cursor to move to button 1, stop the cursor and click. If the cursor does not stop in the button's click-able area, adjust it's position using speech command before clicking the button,
- repeat the clicking task from button 1 to button.8;
- at last click button 1.

All the steps are controlled by speech-based cursor in both types of applications. The application programs automatically record the time, number of stop commands and total number of speech commands issued.

In order to obtain valid data, we give the participants enough time to train the voice recognition engine to ensure it can recognize their speech commands. Before the formal usability test, every participant trained at least 30 tasks to get complete familiarity with the speech-based control solution they would use.

Sears concluded that users had little difficulty accurately select large targets [5], therefore it is only meaningful to measure performance using realistic sized target. There are four sizes for Windows icons: 48 × 48, 32 × 32, 24 × 24, and 16 × 16 pixels[8]. The Windows toolbar displays two sizes for icons: 24 × 24 and 16 × 16 pixels. We chose 3 kinds of square targets measuring 16x16, 24x24 , and 32x32 pixels (referred to as D16,D24,D32). If our speech-based cursor system can work in such testing environment, it will provide an indication of its usefulness for normal windows applications. These three target sizes were tested separately. For each target size, the buttons are arranged the same way as depicted in Figure.2.

During the execution of every task, our test application recorded the number of "Stop" commands, total number of speech commands except "click", and the time to finish a task.

4.4 Hypotheses

We expect the delay-compensated speech-cursor control to have a significant impact on the user's performance. The user's performance is measured by total number of

speech commands to finish a task, the selection time and number of "stop" commands used. The hypotheses for this experiment were:

H0a: The compensate cursor will not have a significant effect on the total number of speech commands and the time required to finish the test tasks compared to a standard speech controlled cursor.

H0b: The compensate cursor will have a significant effect on the total number of speech commands and the times required to finish the test tasks compared to a standard speech controlled cursor.

H1a: The target size will not have a significant effect on compensate cursor control.

H1b: The target size will have a significant effect on compensate cursor control.

(Note, as reported in [5], target size's effect is significant in normal speech-controlled cursor applications.).

4.5 Results

Means and standard deviations for the tasks including number of "stop" commands, total number of commands (including directional commands) and selection time using compensate and standard cursor control solutions are reported in Table 1.

Table 1. Means and standard deviations (in parentheses) of number of "Stop" commands , number of all voice commands, and selection time(in seconds) for tasks completed using two types of cursor controls and 3 types target sizes

	Compensate cursor			Normal cursor		
	Number of "stop"	Number of all commands	Selection time	Number of "stop"	Number of all commands	Selection time
D16	11.23 (1.59)	30.46 (3.18)	40.84 (2.48)	21 (3.87)	60.38 (10.57)	66.29 (10.80)
D24	9.08 (0.28)	26.46 (0.88)	37.45 (1.11)	14.77 (2.62)	40.38 (5.01)	46.50 (5.89)
D32	9 (0)	26 (0)	35.39 (1.06)	12.15 (1.52)	33.38 (3.60)	38.06 (3.93)

For the total number of speech commands, a one-way analysis of variance (ANOVA) with repeated measures for target size was utilized to assess the effect of cursor type. As we expected, the type of cursor control has a significant effect on the total number of commands required to finish the task ($F(1, 76) = 60.12$, $p < 0.001$), and from figure 3, we can see the total number of commands increased a lot using the normal cursor control when the target size is reduced from D32 to D16, but for delay-compensated cursor control solution the number only increased a little.

For selection time, another ANOVA with repeated measures for target size was utilized to assess the effect of cursor type. The result indicated a significant effect for delay-compensated cursor control type ($F(1, 76) = 29.31$, $p < 0.001$). From figure.4, we can see that when the target size is reduced from D32 to D16, the selection time increased a lot using non-compensate solution, and the compensate solution only increased a little at the same time.

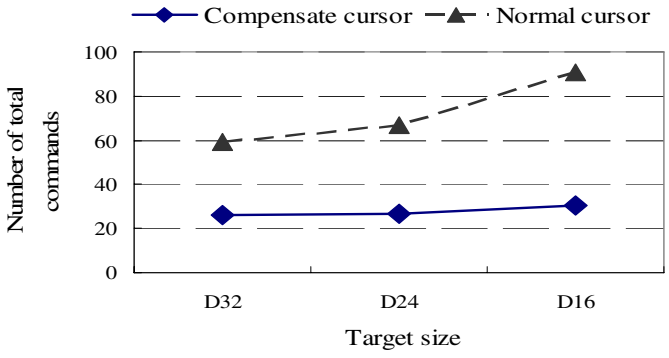

Fig. 3. Means of total commands using compensate and non-compensate solutions

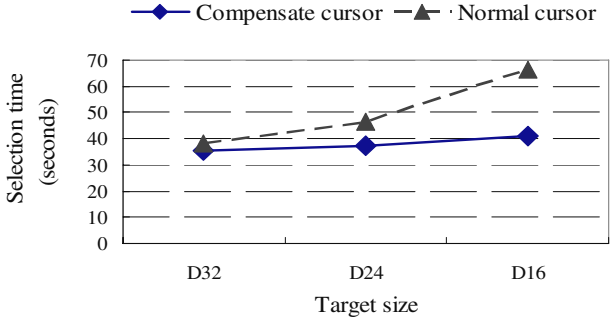

Fig. 4. Means of target selection time using compensate and non-compensate solutions

In addition, the target size has a significant effect on the on the total number required to finish the task for compensate type ($F(2, 36) = 21.59$, $p < 0.001$), the result also indicated that the target size has a significant effect on the selection time for compensate type ($F(2, 36) = 34.57$, $p < 0.001$). But from Fig.3 and Fig.4 can see that the delay-compensated approach fares better: that the size of the target doesn't have as much impact on delay-compensated approach than on normal approach.

H0b and H1b were supported by the data analysis.

5 Discussion

As expected, the delay-compensated cursor control solution provided significant benefits compared with the standard speech-based cursor control solution. When the target size is small the delay-compensated cursor control solution has even more advantage, the total number of commands is only the half of the that of non-compensated solution.

Though target size has a significant effect on the tasks, the effect is far less significant on delay-compensated cursor control solution compared with standard solution.

This is illustrated in Figure.3 and Figure.4, especially when the target size changed from D32 to D24, the target size has no significant effect on total number of commands ($F(1,24)=3.6$, $p>0.05$) for delay-compensated cursor control solution.

For D32 and D24 target sizes, the delay-compensated solution has recorded the least total number of speech commands: 26 times and stop number: 9 times..

We observed that out of the total time to finish the tasks (denoted as Selection Time), the time the cursor took to travel the distance between targets was significant. This time cursor moving time was also constant for different target sizes. Therefore, we considered the number of total speech commands as a more important performance measure that the Selection Time.

Finish the same task uses less time and fewer speech commands means the delay-compensated cursor control solution gives user more efficiency and confidence to use speech-based cursor control.

6 Future Work

Theoretically our solution can compensate the main delays associated with speech recognition, namely speaking time and processing delays. But this approach still can not compensate the reaction delays, for example, the delay introduced when the cursor moves into the target area but the user hasn't reacted immediately, just like a unfocused 100-meter dash athlete hesitates to start to run after the starting gun shot.

We plan to conduct additional future work to investigate the relationship between cursor speed and the reaction delays, and study the way to compensate the reaction delays, or the way to help user to reduce the reaction delays.

Another area for further study is around variable cursor speed. In our current implementation the cursor's speed is constant. We plan to add speed control to the application, so we can control the cursor to move faster when it needs to travel longer distances, and move slower for shorter distances or when it near the targets. We believe such variable cursor speed controls will improve usability and are more realistic.

7 Conclusion

We presented a new delay-compensated solution for speech-based cursor control, where the cursor movement is reversed at the end of the speech command recognition to compensate speech delay. We conducted preliminary usability tests to show that delay-compensated cursor control provides the expected benefits. The result is encouraging, compared with the normal speech-based cursor, our solution allows users to finish the same task faster and use fewer commands.

Using delay-compensated cursor control solution can help speech-based cursor control systems to overcome the limitation of accurately positioning control associated introduced by the recognition delay. At the same time the user can control the cursor by a natural way of using the voice.

Acknowledgements. This material is based upon work supported by the Science Foundation of Technology Bureau of HuangShi, Hubei Province in P.R. China (Grant No: HZT [2007]40), Any opinions, findings and conclusions, or recommendations

expressed in this material are those of the authors and do not necessarily reflect the views of Technology Bureau of HuangShi. We also supported by the plan for scientific and technological innovation team of excellent young and middle-aged in institute of high learning of Hubei Province in P.R. China (Grant No:T200806).

References

1. Olwal, A., Feiner, S.: Interaction techniques using prosodic features of speech and audio localization. In: IUI 2005: Proc. 10th Int. Conf. on Intelligent User Interfaces, pp. 284–286. ACM Press, New York (2005)
2. Harada, S., Landay, J.A., Malkin, J., Li, X., Bilmes, J.A.: The vocal joystick: evaluation of voice-based cursor control techniques. In: Proc. Assets 2006, pp. 197–204. ACM Press, New York (2006)
3. Dai, L., Goldman, R., Sears, A., Lozier, J.: Speech-based cursor control: a study of grid-based solutions. In: Proceedings of the 6th international ACM SIGACCESS conference on Computers and accessibility, vol. (77-78), pp. 94–101. ACM Press, New York (2004)
4. Igarashi, T., Hughes, J.F.: Voice as sound: using non-verbal voice input for interactive control. In: UIST 2001: Proceedings of the 14th annual ACM symposium on User interface software and technology, pp. 155–156. ACM Press, New York (2001)
5. Sears, A., Lin, M., Karimullah, A.S.: Speech-Based Cursor Control: Understanding the effects of target size, cursor speed, and command selection. Universal Access in the Information Society 2(1), 30–43 (2002)
6. Mihara, Y., Shibayama, E., Takahashi, S.: The migratory cursor: accurate speech-based cursor movement by moving multiple ghost cursors using non-verbal vocalizations. In: Assets 2005: Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility, pp. 76–83. ACM Press, New York (2005)
7. Sporka, A.J., Kurniawan, S.H., Slavík, P.: Whistling user interface (U3I). In: User Interfaces for All, p. 472 (2004)
8. Windows User Experience Team Microsoft Corporation (July 2001). Creating Windows XP Icons, `http://msdn.microsoft.com/en-us/library/ms997636.aspx` (visited, July 2008)