Non-speech Input and Speech Recognition for Real-time Control of Computer Games

Adam J. Sporka[1], Sri H. Kurniawan[2], Murni Mahmud[2], Pavel Slavík[1]

Czech Technical University in Prague
Department of Computer Science and Engineering
Karlovo nám. 13, 12135 Praha 2
Czech Republic
Tel.: +420-22435-7440

sporkaa@fel.cvut.cz
slavik@fel.cvut.cz

University of Manchester
School of Informatics
PO Box 88, Manchester M60 1QD
United Kingdom
Tel.: +44-161-306-8929

s.kurniawan@manchester.ac.uk
M.Mahmud-2@postgrad.manchester.ac.uk

ABSTRACT

This paper reports a comparison of user performance (time and accuracy) when controlling a popular arcade game of Tetris using speech recognition or non-speech (humming) input techniques. The preliminary qualitative study with seven participants shows that users were able to control the game using both methods but required more training and feedback for the humming control. The revised interface, which implemented these requirements, was positively responded by users. The quantitative test with 12 other participants shows that humming excelled in both time and accuracy, especially over longer distances and advanced difficulty levels.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces – *Input devices and strategies (e.g., mouse, touchscreen), Interaction styles (e.g., commands, menus, forms, direct manipulation), Voice I/O.* H.1.2 [Models and Principles]: User/Machine Systems – *Human factors*. I.3.6 [Computer Graphics]: Methodology and Techniques – *Interaction techniques.*

General Terms

Measurement, Experimentation, Human Factors.

Keywords

Acoustic input, game control, voice interaction, non-speech control, speech recognition, motor-impaired users.

1. INTRODUCTION

The need for making computers more accessible to people with special needs, including people with motor impairment, has been a hot topic of the past decade, with some high profile accessibility initiatives such as WCAG, Section 508 of the Rehabilitation Act in USA, and the EU's eAccessibility initiative, to name a few.

ASSETS'06, October 22–25, 2006, Portland, Oregon, USA.

The increasing participation of the community of computer users with special needs consequently brings about an increased demand of involving this community in the area of computer-supported entertainment, including the computer games. This brings about the need for understanding the benefits and drawbacks of different assistive input devices and techniques that may be used for playing computer games.

The user interfaces of most of the turn-based[1] or strategic games (e.g. chess, checkers, or SimCity) do not depart far from the paradigm of common desktop applications. While these games may involve a large number of operations, they usually do not expect users' actions in timely manner, and therefore are not likely to disadvantage users with motor impairments.

However, in arcade games[2] (e.g. Pitfall! [1] or Tetris [12]), people with motor impairments are usually at a disadvantage when playing against people without motor impairments due to their disabilities: Even though only a limited number of game commands is required, these commands have to be issued rapidly – and many assistive devices are unable to cope with this problem.

Motor impairments are a loss or limitation of function in muscle control and/or movement, or a limitation in mobility. There are various aids that can help the people with motor impairments use interactive systems more effectively, including tools such as Sticky Keys that make difficult key combinations more accessible, voice recognition systems, pointers controlled by eye, mouth or head movements, and text entry systems to help enter messages with fewer keystrokes [20]. Whilst most of these devices have been proven to successfully help these users, some of them incur high cost, which made acoustic input, such as speech recognition systems, a popular choice [4].

Speech recognition is suitable for many applications, in which the interaction with a computer system is based on a dialogue between users and computers (a command—response style of interaction), ranging from dictation of text to various telephony service applications [3]. Some people with motor impairment use adapted speech recognition technology to replace a mouse, e.g. by

[1] Turn based games are games where players are allowed a period of analysis before committing to a game action, ensuring a separation between the game flow and the thinking process [19].

[2] Arcade games are games with fast action where hand-eye coordination is the primary skill needed to beat the game [7].

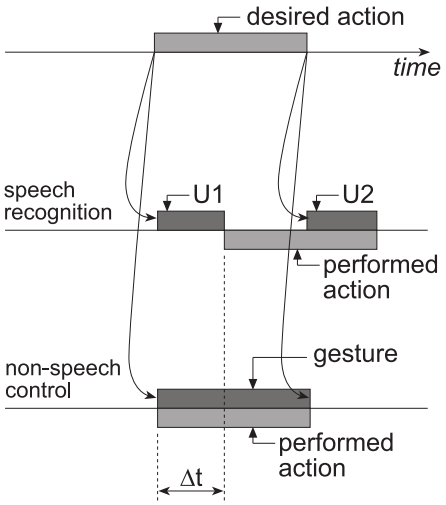

Figure 1. Theoretical difference Δt between the speech and non-speech control response time.

issuing directional commands such as 'LEFT', 'UP' [8], or iteratively selecting square areas on screen [4].

Recently, *non-speech input* (the use of other sounds than speech that are produced by the human vocal tract, such as humming, hissing, or whistling, for controlling user interfaces) is gaining popularity within the HCI community. These sounds can be characterized by a variety of features (such as pitch, volume, or timbre) that may be directly used as an input to user interfaces.

Non-speech input has been reported and evaluated in many different contexts. The entry of values in a voice-operated remote control of a TV set is proposed in [7]. A program described in [21] was used to launch predefined UNIX commands upon reception of corresponding melodies. Also, the non-speech input has already been successfully employed to emulate standard mouse [2, 17, 19] and keyboard [16].

Speech recognition and non-speech acoustic inputs differ in the following two aspects:

1) The response delay. While most speech recognizers wait for users to finish their utterance before initiating the recognition, the elements of non-speech audio input can be processed and interpreted on-the-fly as soon as users produce them and thus shorten the response delay of the processing.

This is demonstrated on Figure 1. In speech recognition systems, utterances for the start and end of a desired action (U1, U2) must be completed in order to begin and end a desired action. In non-speech control, a desired response may be mapped directly to a single input sound and being executed while the tone is held.

2) The domain size. With a good design of grammatical mapping of the utterances, speech recognition may be used to trigger a wide range of operations, wider than is possible with direct application of non-speech input. There is only a limited amount of features of the non-speech sounds that may be extracted and assigned to different input channels.

Because of these differences, speech and non-speech inputs are suitable for different genres of computer games. We can argue that turn-based or strategic games are perhaps best controlled using speech recognition (as they are not time-sensitive and may require a large number of operations) while non-speech control

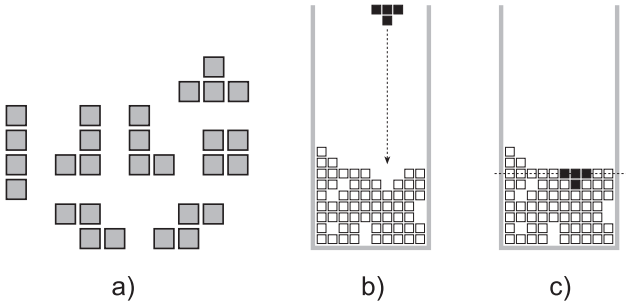

Figure 2: The basic concepts of Tetris. a … possible tetrominos, b … a descending tetromino, c … completion of a row.

may be more suitable for arcade games, which rely heavily on real-time control. The use of non-speech control in arcade games has already been described in past studies, for example in [6], and demonstrated in [15].

2. THE STUDY OUTLINE

This paper presents a comparative study of speech recognition and non-speech input, employed to control arcade games. We decided to perform this comparison on Tetris [10, 12] for its simplicity and popularity, and for being a game that is largely affected by the performance of the input method. Tetris is indeed a good representative of the arcade genre and it is still one of the most popular games [5].

There is one caveat of this study. We did not include people with motor impairment in this study as the aim of this study was to provide early feedback on the evaluation methodology and user interface that we developed.

This study aims to answer one fundamental question: which of the two input methods, speech recognition or non-speech input, would yield a better performance? We also decided to include the keyboard control into our study as a baseline reference.

There are many variants of Tetris; however, the basic rules are common for all. The player is required to solve a sequence of episodes. In each episode [10], a presented tetromino—randomly selected from seven possible tetrominoes, see Figure 2a—must be maneuvered during its descent (shifting the tetromino to the sides or rotating it) from its spawning position to a desired position on the top of previously placed tetrominoes in the playing field which is a vertical matrix of square cells (Figure 2b). The dimensions of the playing field are usually 10 by 20 to 22 cells.

Eventually, the tetromino lands in the bottom of the field or on the top of the previously landed tetrominoes. When a row is completed, it is removed, all rows above it are shifted towards the bottom, and the player receives a bonus score.

The difficulty of the game is determined by the speed of the descent, which in most implementations increases over time as the game progresses.

The game ends when there is not enough space for spawning a new tetromino. The objective of the game is to stay in the game for as long as possible and thereby maximize the score.

For this study, we used our own implementation of Tetris to employ custom input methods for controlling the game as well as to create output log files.

2.1 The Input Methods

This section describes the input methods we used for our experiments.

2.1.1 Keyboard Control

Keyboard was the input device in the original Tetris [12]. The following set of control keys is in use in many implementations of the game and was used in our implementation: the cursor keys of ← and → are for movements to the left and right respectively, the ↑-key is for counter-clockwise turn and the ↓-key is for speeding up the landing. In the subsequent text, we shall refer to these commands as *left*, *right*, *turn*, and *drop*.

2.1.2 Speech Recognition

We used the industry standard MS SAPI 5.1 library to implement the speech recognition input in our system. In all of our tests we used the default settings of the speech recognition engine. To issue a command, the users have to utter its name, i.e. to say "left" for triggering the *left* command, and so on.

First, we defined the commands *left* and *right* to initiate the movements in the relevant direction (which corresponds to depressing a cursor key). We introduced a new command *stop* to stop the movement at desired position (= releasing the key). A brief test, however, revealed that such interaction was not feasible due to significant delay of recognition (about 0.5 sec).

We have therefore redefined the *left* and *right* commands to move the piece only by one cell in given direction, as a metaphor of hitting the cursor key. The commands *turn* and *drop* retained their meaning as described in the previous paragraph.

2.1.3 Non-speech Control

As mentioned in the Introduction, the non-speech control is based on an application-dependent interpretation of non-speech sounds produced by the user.

A typical organization of the signal processing pipeline for a non-speech input is shown on figure 3. The sound signal acquired by microphone (and a sound card) is processed by a sound analyzer. Extracted features (such as the presence of a tone and the pitch of the tone) are being tracked by gesture recognizer which identifies the non-speech gestures in the incoming signal.

Non-speech gestures are understood as short melodic patterns of defined pitch profile and/or length. Depending on the context in which the non-speech input is used, the set of gestures for individual commands or operations is defined to cover all possible operations. The description of gestures may be also based on different features, such as timbre or volume, as used for example in [2].

The results of our previous studies [16, 19] indicated a viability of non-speech input for controlling a mouse cursor as well as emulating a keyboard. Our studies also revealed that humming ("*mmm*") was the non-speech sound that most participants reported to be easy and comfortable to produce even over an extended period of time. We have therefore decided to use humming for the non-speech control in this study.

The gestures used in our implementation are shown in Figure 4. A tone with falling pitch has been assigned the command *left*. A tone with raising pitch is the gesture for the command *right*. To eliminate the influence of unintended minor fluctuations, the pitch of tone must fall or rise by at least a minimum increment of about two semitones from the initial pitch to be valid falling or raising tone. As long as the tone is held, the current tetromino is moved to

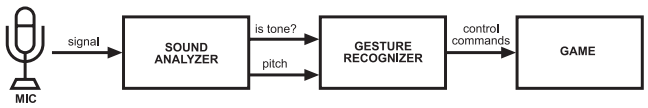

Figure 3: Non-speech sounds processing pipeline

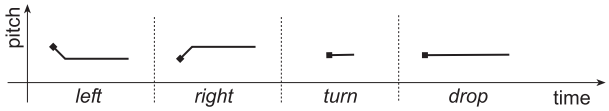

Figure 4: Pitch profiles of the set of gestures used in our implementation of Tetris.

the left or right. The tetromino stops once the tone disappears (the inherent *stop* command).

The remaining two commands, *turn* and *drop* were mapped to flat tones of different length. The *turn* command is issued upon recognition of a flat tone shorter than about 0.3 s. Any longer flat tone triggers the *drop* command upon its termination.

The non-speech sound interpreter can be modeled using a state automaton which is schematically shown in Figure 5. An actual sequence of gestures, as recorded during a game, is shown in Figure 6.

For measurement of the pitch of humming, we have used autocorrelation, as described for example in [14].

3. A PILOT STUDY

For the pilot study, we used our implementation of the game in which only the essential functionality was included. However, the layout of the game screen, as shown in Figure 7, was not different from many other Tetris applications. We implemented the game as a standalone application for the win32 platform.

The aim of the qualitative pilot study was to identify potential weaknesses of the user interface and to suggest a suitable methodology for the subsequent quantitative test.

Seven users (3 M, 4 F, average age = 15.1, S.D. = 2.21 years old) were invited to participate in the study. They all were students of a high school in Manchester area. They had no previous experience with speech recognition or any application that uses

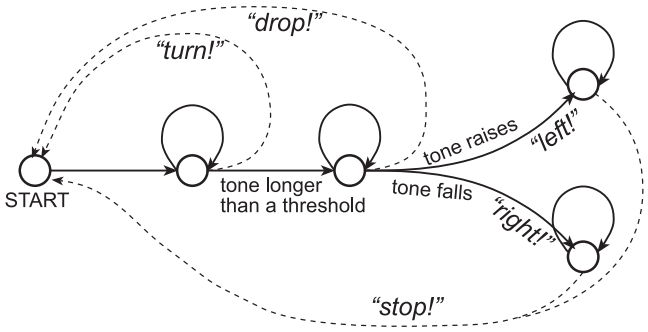

Figure 5: Gesture recognizer. Solid lines – transitions when tone exists; dotted lines – transitions upon silence; captions in quotes – control commands, lowercase captions – transition conditions.

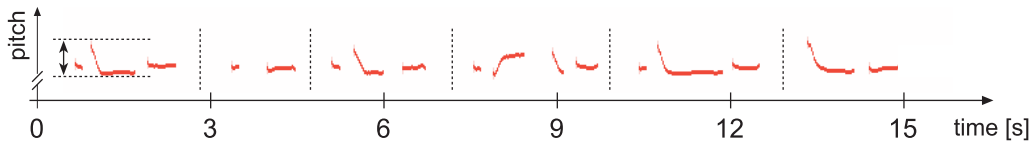

Figure 6: A sequence of gestures recorded during an actual game, shown in a correct time scale. The height of the vertical arrow by the left side of the chart corresponds to interval of three semitones. The vertical dashed lines separate the episodes of the game.

non-speech input. However, they were familiar with at least one commercial implementation of the game and they previously had spent at least several hours playing it.

During the individual sessions, each participant was asked to learn to play the Tetris using speech commands and humming. At their disposal, they had a recording of the control sounds produced by an expert user (the developer of our implementation). They were given 15 to 30 minutes to get acquainted with the acoustic input methods. After that, they were asked to play the game using keyboard, non-speech input, and speech recognition for five minutes. Their play was observed by an experimenter.

Overall, the participants reported that playing the game through keyboard was the easiest because of their previous experience with the game. They were not able to memorize the acoustic gestures for humming. They reported to be much more confident when using the speech recognition, however they were convinced that they would eventually be able to learn the correct non-speech gestures if given more time to do so.

This qualitative study had resulted in two very useful suggestions for the next round of design and evaluation:

- The participants should be given more time for training of the humming control before measuring their performance. An integrated training application should be implemented in the next version of the user interface, and the participants should be asked to perform training before the actual performance measurement in the quantitative test.
- The participants required some form of visual feedback of their humming. A visual feedback of the pitch of the produced humming as well as the visualization of the recognized non-speech gestures would be included in the revised application.

Based on the findings in the qualitative study, the implementation of the game was modified. Especially, we added the non-speech control feedback gauges that allow users to see the pitch profile of their voice and corresponding gestures as identified by the gesture recognizer (see Figure 8).

4. QUANTITATIVE TESTS

To provide quantifiable evidence on the difference (or lack of) between the effectiveness of the humming and speech control, two tests were performed: lateral speed test and episode test. They are described in detail in Section 4.1 and 4.2. At the end of these two tests, the participants were asked to play Tetris using both input methods and to provide comments and feedback.

Twelve people who had some experience playing Tetris participated in the test. Unfortunately, the test was not gender-balanced: 11 of them were men. Their average age is 26.9 years old (S.D. = 4.66 years).

All participants were given a copy of our program and asked to learn to play Tetris using speech recognition and non-speech input at home before performing the tests in our lab. In average they spent 35 minutes on training.

4.1 Lateral Speed Test

For the purposes of this study, we defined the following terms:

- The *preset lateral speed* is the distance measured in the number of cells the tetromino travels per second horizontally

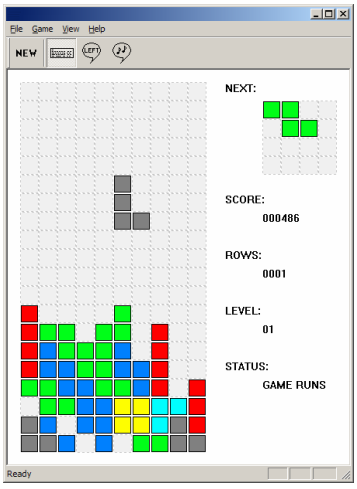

Figure 7: A screenshot of the first version of our implementation.

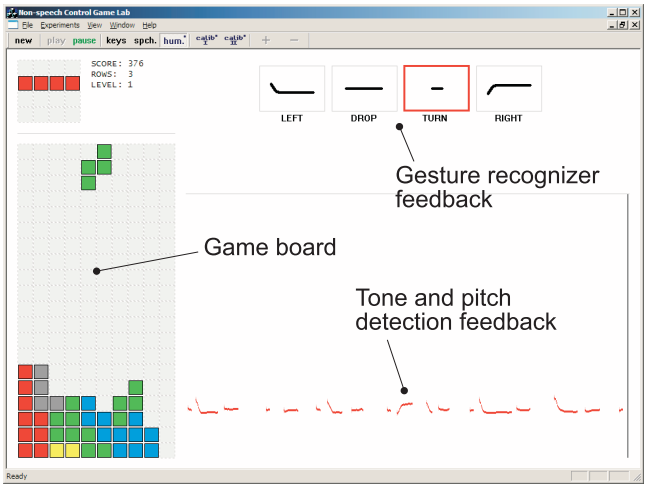

Fig 8: The revised user interface.

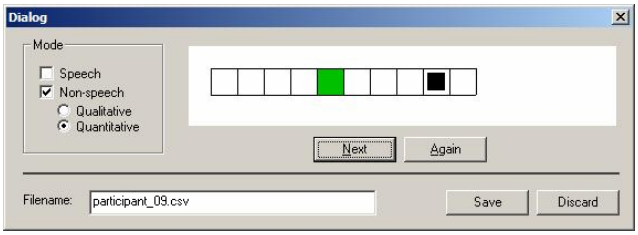

Figure 9: Lateral speed test screenshot

as long as the *left* or *right* non-speech gesture is held. The preset speed is not defined for the speech recognition as in our scenario, the speech commands *left* and *right* make the tetromino move only one cell sideways at a time (see section 2.1.2).

- The *effective lateral speed* is the number of cells the tetromino travels horizontally (the distance between the origin and the destination cells) divided by the actual time the player needs to complete the desired motion.

The purpose of this study was to examine the effective speed in two different input methods, the speech recognition and non-speech input. An average effective speed was measured for both methods. Consequently, the relationship between the preset and corresponding effective speeds was investigated for the non-speech input method to determine the preset speed at which the users would gain the highest effective speed.

In this test, we simulated the horizontal steering operation in Tetris without requiring the participants to think of any strategy commonly performed in the game – kind of a Fitts' Law test without target's size variations.

Figure 9 shows the stimulus used for this test. The users were asked to simply move the small black square from its origin to the target position coloured green (the larger shaded square). Pressing the *Next* button records the current trial and advances to the next trial. Pressing the *Again* button resets the trial and returns the box to its origin. This could be used in case of unexpected disturbance in that particular trial. The data gathered were saved to a file for further processing.

4.1.1 Speech Recognition Test

In this test, users were supposed to use the speech commands '*left*' or '*right*' to move the black square at their own pace. The test consisted of 16 trials (distances 1, 2, 4, and 6 cells; 2 directions; 2 repetitions). The effective lateral speed was calculated for each trial as the distance divided by the time between the start of the first utterance of a command and the recognition of the last command in trial.

Figure 10 depicts the distribution of the effective speeds of all participants. The average effective speed across all participants is 1.4 blocks per second (S.D. = 0.49), and its distribution is close to normal.

T-test shows that across all subjects, the values of the average effective speed for the motion to the left and right were not significantly different ($p>0.05$). We may imply that the subjects were equally capable of controlling the movements to the right and to the right using speech.

One-way ANOVA shows that the average effective speed decreases significantly with the increase of distance traveled ($F_{3,159} = 32.1$, $p = .000$). The LSD post-hoc analysis reveals that the difference is only insignificant between the 4-cell and 6-cell distances. This can be interpreted as the fact that in our scenario, the speech is more effective for controlling cursor movement at short distances (i.e. 1- or 2-cell movement) than at longer distances.

4.1.2 Non-speech Input Test

In the non-speech input test, the effective speed was measured for five different preset speeds, as summarized on the left side of

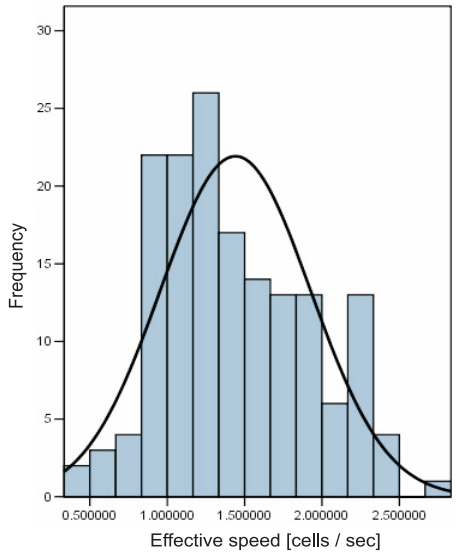

Figure 10: Speech-controlled effective speed distribution

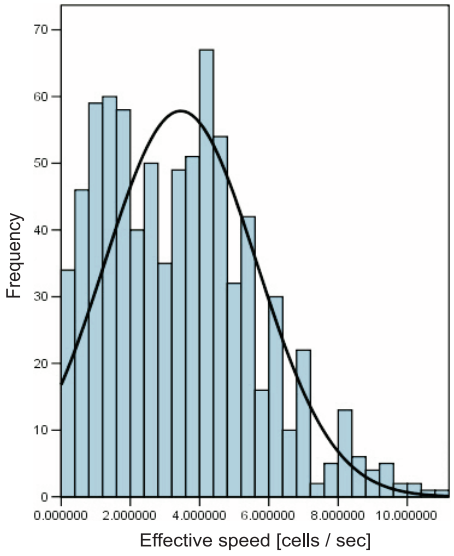

Figure 11: Humming-controlled effective speed distribution

Table 1: Effective speeds at various preset speeds

Preset Speed		Effective Speed		
Code	Magnitude [cells/s]	N	Mean [cells/s]	S.D.
S3	14.2	159	3.0	2.70
S4	10.8	159	3.8	2.72
S6	7.35	159	3.9	2.07
S8	5.4	160	3.5	1.62
S10	4.35	159	3.1	1.43
Total		796	3.5	2.20

Table 1. Lower preset speed was expected to yield less overshoots but lower effective speed and vice versa. In total, each participant performed 80 trials (16 trials as described in previous section × 5 preset speeds).

Figure 11 depicts the distribution of the effective speeds of all participants for all preset speeds. As the figure shows, the average effective speed across all participants was 3.5 cells per second (S.D. = 2.20), around 2.5 times faster than the speech-controlled effective speed. The distribution is skewed to the left indicating that there is a tail of expert performance in this task.

The T-test showed that, similar to the case of speech test, across all subjects, the values of the average effective speed for the motion to the left and right were not significantly different ($p>0.05$).

The one-way ANOVA showed that the averages of the effective speeds for different preset speeds varied significantly ($F_{4,795} = 4.852$, $p = .001$). Table 1 and Figure 12 show the averages of the effective speed at different magnitudes of the preset speed. As can be seen, there was not a linear relationship between the preset speed and effective speed. S4 and S6 preset speeds yielded significantly better effective speeds than other preset speeds (as also verified using LSD post-hoc analysis).

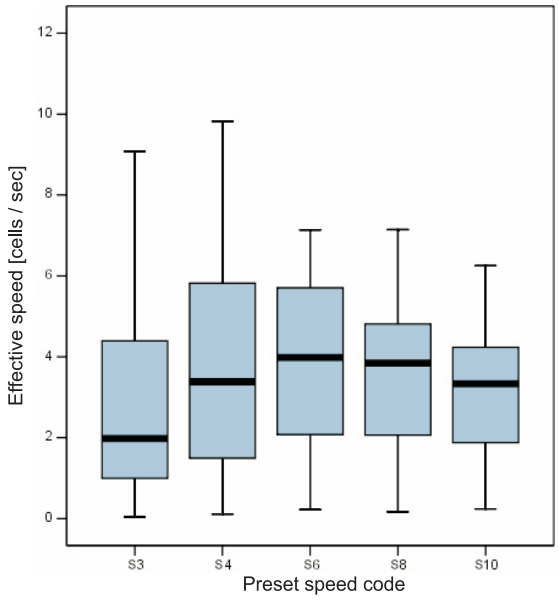

Figure 12: Visualization of the data contained in Table 1

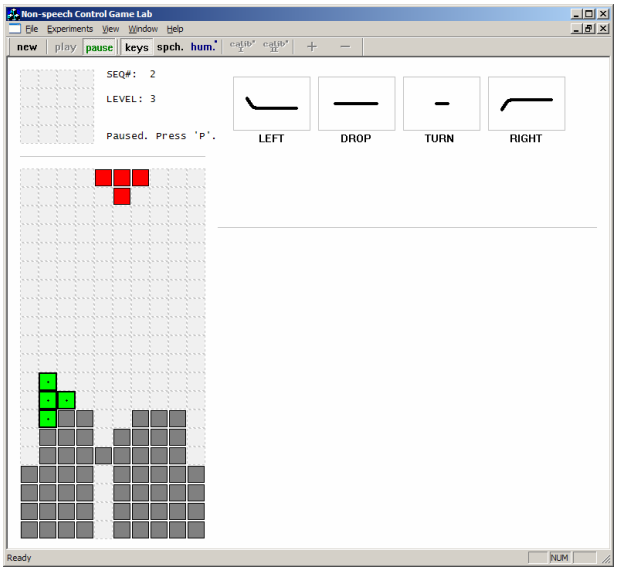

Figure 13: A screenshot of the episode test mode.

One-way ANOVA also showed that the averages of the effective speeds for different distances were significantly different. LSD post-hoc analysis showed that the effective speeds were higher at bigger distances and that the effective speeds for 1 and 2 cells movements were not significantly different. We find this very interesting, as this implies that controlling movements through humming results in faster movements over longer distances.

The preset speed of 7.35 cells per second (S6) yielded the maximum mean effective speed across all players.

4.2 Episode Test

Episode test aimed purely at testing the accuracy of speech and humming input. To perform this test a screen similar to the Tetris screen was used, as depicted in Figure 13.

To rule out the variance of individual gaming strategies, the test consisted of sequence of predefined episodes repeated in 3 difficulty levels differing in tetromino descent rates (1.25, 2.0, and 4.5 cells/sec., referred as slow, medium, and fast respectively from here on).

For each episode, the initial configuration of the board as well as the initial and target positions of the piece were defined. The participant's task was to manipulate the presented tetromino so that it lands in the given target position, as shown in Figure 14. The same sequence of episodes was to be performed by humming, speech recognition, and—as an accuracy baseline—also using keyboard which the participants would have been very familiar

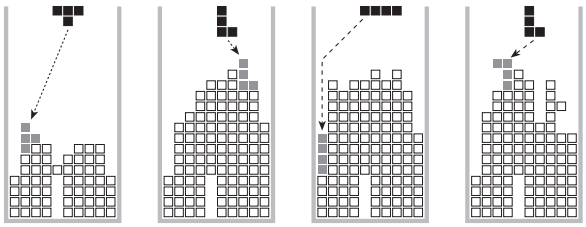

Figure 14: An example of some of the episodes used in the test.

Table 2: Accuracy of keyboard, speech, and non-speech control at three different descent rates.

Subj.	Keyboard			Humming			Speech		
	slow	med.	fast	slow	med.	fast	slow	med.	fast
1	1	1	0.92	1	0.5	0.83	0.67	0.5	0.17
2	1	1	0.92	0.92	0.83	0.25	0.67	0.67	0.25
3	1	1	1	1	1	0.67	0.83	0.67	0.17
4	1	1	0.83	0.83	0.75	0.25	0.75	0.42	0.17
5	1	1	0.92	0.92	0.92	0.42	0.92	0.42	0.17
6	1	1	1	1	0.75	0.83	0.58	0.58	0.17
7	1	1	0.83	0.83	1	0.42	0.58	0.42	0.17
8	1	1	0.83	1	0.83	0.42	0.58	0.42	0.17
9	0.92	1	1	0.58	0.67	0.17	0.58	0.42	0.08
10	1	1	1	0.67	0.58	0.33	0.83	0.58	0.17
11	1	1	0.83	1	0.83	0.5	0.75	0.58	0.25
12	1	1	0.67	0.5	0.75	0.33	0.5	0.42	0
Mean	0.99	1	0.9	0.9	0.8	0.5	0.7	0.5	0.2
S.D.	0.02	0	0.1	0.18	0.15	0.22	0.13	0.1	0.07

with. The order of this test of these three input methods was balanced across the 12 participants.

Accuracy was calculated as the number of correctly landed tetrominoes divided by the total number of episodes. Table 2 shows the accuracies for all input methods and speeds for all of the 12 participants. Figure 15 shows the accuracy averages of the 12 participants for the three difficulty levels.

The accuracy declined with the increasing level of difficulty in all input methods. The keyboard produced the best results (as expected), followed by the humming and speech recognition. Using t-tests, we verified that in all three levels the humming input produced significantly more accurate movements than speech input did, with $p<0.05$. The interaction between the three acoustic methods was not significant.

The difference in accuracy is a consequence of different responsiveness of the three methods. The game is controlled

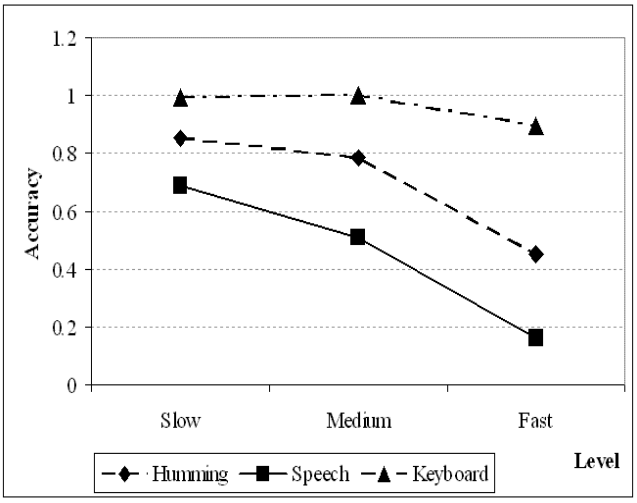

Figure 15: Accuracy averages using speech and humming for the three difficulty levels.

fastest using the keyboard, followed by non-speech control, and slowest by the speech recognition. In many episodes, the users simply did not have enough time to perform all necessary maneuvers when using speech recognition.

If we could interpolate the result to other speeds that we haven't tested, we can argue that the result indicates that humming provides a more precise control than speech.

5. PRELIMINARY USER EVALUATION OF THE TETRIS GAME

After the participants finished the two performance tests, they were asked to play Tetris using both input methods and provide comments and feedback. The actual games were not measured as we would have to tease apart the effects of expertise/strategy and the effectiveness of the input methods. In general, the participants enjoyed the games with both acoustic methods, but less so when using speech recognition at higher difficulty levels. The participants were confident that they would be able to achieve high scores (albeit perhaps not as high as when playing using keyboard) with humming. They also commented that this application would be very entertaining and highly appreciated by the community of people with motor impairment.

Some participants suggested that more perceptible feedback should be produced by the system to help them distinguish between the *stop* and *turn* non-speech gestures. Sometimes they produced a *stop* command instead of a *turn* command as they mistakenly felt they already held the tone for long enough. In response, we included acoustic feedback to the game. As soon as the length of the tone would exceed the threshold between these two gestures (see Figure 4), a short soft click would be played by the system. This modification was positively commented by the participants.

6. CONCLUSION AND FUTURE WORK

This study aimed at comparing user performance when using two different acoustic input methods: speech recognition and non-speech input by humming. To provide a context of this measurement, a popular arcade game Tetris was used.

This study started with a qualitative evaluation of the Tetris game application we developed. The seven participants were able to use both acoustic input methods to play the game, but requested more training and some feedback about the humming operation.

The revised Tetris game interface implemented these two suggestions, and the resulting game was positively responded by the participants of the user evaluation.

The quantitative test showed that whilst speech could be used to control movement, its performance degraded when the distance traveled was high. On the contrary, humming performance was better when traveling over long distance. In average, humming-controlled movements were around 2.5 times faster than speech-controlled movements.

Humming was also up to three times more accurate than speech in controlling tetrominoes in the episode mode, with the difference being more pronounced when the tetrominoes' descent rates were higher. An inclusion of acoustic feedback that helped distinguish between the gestures of the same pitch profile but different length was positively accepted by the players.

In summary, this study showed that the non-speech input was a more effective way for controlling Tetris than the speech

recognition. This may indicate that the non-speech input is more suitable for control of games requiring rapid yet accurate responses of the players.

There were several limitations of this study, which opens avenues for further work. Firstly, we have to think of a way of comparing the performances of the actual Tetris game using these two input methods, taking into account the individual differences in Tetris expertise and the random occurrence of certain tetrominoes.

Secondly, a longitudinal study would be an interesting continuation of this study as it will allow us to observe users' learning curves.

Finally, as some of the users that can benefit the most this system are people with motor impairments, we should extend the study with a formal usability study with this user group.

7. ACKNOWLEDGMENTS

This work has been partly supported by the Ministry of Education, Youth and Sports of the Czech Republic under the research program LC-06008 (Center for Computer Graphics).

Tetris is currently a registered trade mark of The Tetris Company based in Hawaii, USA. Tetris was originally invented by Alexey Pazhitnov.

8. REFERENCES

[1] Activision, Inc. *Pitfall!* A video game for Atari 2600, 1982.

[2] Bilmes, J. A., Li, X., Malkin, J., Kilanski, K., Wright, R., Kirchhoff, K., Subramanya, A., Harada, S., Landay, J. A., Dowden, P., and Chizeck, H. The vocal joystick: A voicebased human-computer interface for individuals with motor impairments. In *Proceedings of Human Language Technology Conference* (Vancouver, Canada). 2005.

[3] Cohen, M. H., Giangola, J. P. and Balogh, J. *Voice User Interface Design*. Addison-Wesley, 2004

[4] Dai, L., Goldman, R., Sears, A., Lozier, J., Speech-Based Cursor Control: A Study of Grid-Based Solutions, In *Proceedings of The Sixth International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2004*, Atlanta, USA, ACM, 2004.

[5] Gawley, R.E. Chirality Made Simple: A 1- and 2-Dimensional Introduction to Stereochemistry. *Journal of Chemical Education*, 82, 7 (Jul, 2005), 1009-1012

[6] Hämäläinen, P., Mäki-Patola, T., Pulkki, V. and Airas, M. Musical computer games played by singing. In *Proceedings of The 7th International Conference on Digital Audio Effects* (Naples, Italy). 2004, 367–371.

[7] Igarashi, T., and Hughes, J. F., Voice as sound: using non-verbal voice input for interactive control. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*. ACM Press, New York, 2001, 155-156.

[8] Karimullah, A.S., Sears, A. Speech-Based Cursor Control, In *Proceedings of The Fifth International ACM SIGCAPH Conference on Assistive Technologies, ASSETS 2002*, Edinburgh, UK, ACM, 2002.

[9] Makegames.com. Glossary. (Jun, 1 2006); http://www.fastgraph.com/makegames/glossary.html.

[10] Maglio, P. P. , and Kirsh, D. Epistemic Action Increases With Skill. In *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society*. Lawrence Erlbaum, Mahwah, NJ, 1996, 391-396.

[11] Noyes, J. M., Frankish, C. R. Speech recognition technology for individuals with disabilities. *Augmentative & Alternative Communication 8*, 4, 1992, 297-303.

[12] Pazhitnov, A., Gerasimov, V., Pavlovsky, D. *Tetris. A video game. Academy of Sciences*. Moscow, Russia, 1985.

[13] Persson, M. *Development of three AI techniques for 2D platform games.* Ph.D. Thesis, Department of Computer Science, Karlstad University, Sweden, 2005.

[14] Rabiner, L. R. On the use of autocorrelation analysis for pitch detection. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-25(1):24–33, February 1977.

[15] Shahar, E. *The PAH! game*. (May, 29 2006); http://www.designer.co.il/pah/.

[16] Sporka, A. J., Kurniawan, S. H., and Slavik, P. Non-speech Operated Emulation of Keyboard. In *Proceedings of the Cambridge Workshop on Universal Access and Assistive Technology, CWUAAT 2006. Designing Accessible Technology.* Springer-Verlag, London, 2006, 145-154.

[17] Sporka, A. J., Kurniawan, S. H., and Slavik, P. Acoustic control of mouse pointer. *Universal Access in the Information Society*, 4, 3 (Mar. 2006), ISSN 237-245.

[18] Sporka, A., Kurniawan, S.H., Slavik, P., Mahmud, M. Tonal Control of Mouse Cursor: A Usability Study with The Elderly. In *Proceedings of HCI International 2005*, *Las Vegas*, 25-27 July, 2005, CD-ROM.

[19] Sporka, A. J., Kurniawan, S. H., and Slavik, P. Whistling User Interface (U3I). *8th ERCIM International Workshop "User Interfaces For All", LCNS 3196* (Vienna, June 2004). Springer-Verlag, Berlin Heidelberg, 2004, 472-478.

[20] Usability First. Accessibility: Types of Accessibility Aids. (Jun, 1 2006); http://www.usabilityfirst.com/accessibility/types.txl.

[21] Watts, R., Robinson, P. Controlling computers by whistling. In *Proceedings of Eurographics, Cambridge, UK*, 1999.

[22] Wikipedia. Turn-based strategy — Wikipedia, The Free Encyclopedia (Jun, 1 2006); http://en.wikipedia.org/wiki/Turn-based_game.