Adaptation of a Low Cost Motion Recognition System for Custom Operation from Shrink-Wrapped Hardware

Stamatios Sotiropoulos Kyprianos Papademetriou Apostolos Dollas

Microprocessor and Hardware Laboratory
Electronic and Computer Engineering Department
Technical University of Crete
Chania 73100, Greece

sotirop@mhl.tuc.gr kpapadim@mhl.tuc.gr dollas@mhl.tuc.gr

ABSTRACT

A low cost, embedded, reconfigurable device for motion detection of kinetically challenged persons has been developed. This paper presents the methods by which the device can be adapted to individual user's needs. The accelerometer data inputs are processed by the determination of the dominant axis of each motion, and by the determination of thresholds and delays that can be used to customize the system to individual needs. The resulting system is a highly adaptable input device which maintains the low cost constraints of less than $70.

Categories and Subject Descriptors

B.7.1 **[Integrated Circuits]**: Types and Design Styles – a*lgorithms implemented in hardware, microprocessors and microcomputers,* C.3 **[Computer Systems Organization]:** Special-Purpose and Application-Based Systems – *real-time and embedded systems, signal processing systems,* H.5.2 **[Information Interfaces and Presentation]**: User Interfaces – *evaluation/methodology, input devices and strategies, prototyping, training help and documentation,* I.5.4 **[Pattern Recognition]**: Applications – *signal processing, waveform analysis,* K.4.2 **[Computers and Society]**: Social Issues – *assistive technologies for persons with disabilities, handicapped persons/special needs.*

General Terms

Algorithms, Measurement, Design, Human Factors.

Keywords

Embedded, Reconfigurable, Motion Detection, I/O device, Kinetically Challenged.

1. INTRODUCTION

The problem of motion detection and recognition has been considered from a number of perspectives, ranging from I/O for virtual reality environments to gesture recognition systems. Similarly, the problem of I/O devices for kinetically challenged persons has been addressed from a mechanical design perspective to a brain activity detection perspective. In this paper we present the adaptation and motion detection scheme of a *low cost, embedded* I/O device for kinetically challenged persons. The ultimate purpose of this work is to have *shrink-wrapped hardware* which can be customized and re-trained to individual needs without a re-compilation of the design but through reconfiguration, as needed. The hardware will adapt to the needs of different users through hardware reconfiguration, much like general-purpose computers address different problems through the execution of different programs.

In [1] we presented the architecture of the system, and in [2] we presented an improved model in which the hardware efficiency is greatly improved by means of breakdown of complex motions into simple ones. The contributions of this work are:

- The presentation of the low-complexity DSP scheme for the determination of the dominant axis of each motion, while the secondary axis maintains a limited freedom of motion
- The adaptation of the system to different users' needs without change of the hardware
- The user interface for system training.

The organization of the paper is as follows: Section 2 presents related work, which in general is either PC-based (not embedded) or high cost. Section 3 briefly presents the system architecture (which is fully detailed in [1] and [2]) for completeness. Section 4 presents the motion representation scheme. Section 5 the motion detection, adaptation, and training methods, together with the user interface, followed by Section 6 with experimental results and conclusions.

2. RELATED WORK

Various assistive devices that are based on motion measurement and recognition have been developed for persons with motor disabilities [3], [4], [5]. Many of these devices are related to the control of a wheelchair. An electrical wheelchair that is guided by head movements is presented in [6]. An adaptive color face tracking system is used to determine head movements of the user. A color micro-camera acquires user's face images, which are then

WBMA'03, November 8, 2003, Berkeley, California, USA.

loaded in a PC memory. Specific head movements generate commands that control the wheelchair.

Another technique for guiding a wheelchair records the potential between cornea and retina, during eye movement (EOG-Electrooculography) [7] in order to obtain the gaze direction; this system uses EOG measurements acquired by electrodes placed on the head. Data are processed in a PC, which then guides the wheelchair to the direction the user looks at.

A wireless method of controlling a wheelchair has been implemented in [8]. A head mounted computer unit provides a Graphical User Interface, in which the user is able to navigate by combining head movements and voice commands. After selecting the wanted action, the computer unit transfers the command to the wheelchair through the Bluetooth protocol.

Projects using virtual reality technology have also been developed for rehabiliation purposes [9]. The Gesture Control System is a tracking device that recognizes a predetermined collection of simple hand movements, each of which "fires" more complicated tasks. HumanGlove is a 20-position glove with sensors, which allows accurate recording of hand-joint movements [10]. Each sensor measures data related to a Degree of Freedom of the hand. The glove is connected to a PC, where a program displays an animated hand that mirrors movements of the user's hand. Another project implements a prototype human-computer interface with emphasis on commands expressed as hand gestures [11]. The user is in front of a camera connected to a computer. The camera follows the movements of the hand and performs actions (like controlling the TV, turn on/off the lights, etc.) depending on the state and the motion of the hand.

A camera-based system is also presented in [12]. Markers are attached to the eyelids of the user and a camera captures the movement of these markers, as the user moves his eyelids. The data are sampled by a micro-controller and further processed by a PC. The system determines the vertical displacement of each eyelid.

Aiming at surgeon assistance, an active instrument that contributes to improved accuracy during vitreoretinal microsurgery is described in [13]. The instrument that comprises of accelerometers, rate gyros and actuators, senses and compensates physiological tremor and other unwanted movement. The motion captured by the sensors is processed in a PC in order desired and undesired components to be distinguished. Then the actuators move the tip of the instrument in opposition to the motion of the tremor in order to suppress the unwanted movement.

On a different type of projects using reconfigurable logic [14], a wearable motion analysis system comprises of an Altera Flex10KE FPGA and digital accelerometers [15]. The PWM output of the accelerometers, which are mounted on both corners of an eyeglass frame and the left and right sides of a waist belt, is driven to counters implemented in the FPGA. The latter is connected to an embedded computer system with an Adaptive Systems Pentium-class processor module for further processing. The system is used to assess balance and mobility impairments for monitoring patients' progress.

A computer interface device for handicapped people, which uses an FPGA is presented in [16]. The movement of the head, on which an optical sensor is mounted, is used as a positioning signal of the computer cursor. Head movements are also used as input in an FPGA Wheelchair Controller [17]. A combination of an FPGA and a speed adaptor controls the velocity of the wheelchair, after receiving suitable commands from the user in the form of either head movements or voice commands.

An adaptive integral system for assisted mobility has been developed using various sensors, FPGAs and DSPs[18]. A wheelchair is controlled by various electronic guidance alternatives, which include head movements, voice-commands, EOG commanding, digital joystick and breath-expulsion. An FPGA and accelerometers are also used in a project called "3D Eye Tracking Device" [19], which aims to measure head and eye movements. The FPGA acquires real time sensor data, which are then processed by a computer.

It should be noted that essentially all of the above projects use at least one of the following: (a) Large or multiple FPGAs, (b) PC-class fixed computer resources, or (c) expensive equipment to make all necessary data acquisition and calculations. In our system such approaches would not meet cost, size, and power consumption limits for an embedded application.

3. SYSTEM DESCRIPTION

Our system is based on accelerometers to sense hand motion in 2D space. The general characteristics of the system, defined functionally and not in terms of implementation are: (a) low cost solid state (non-mechanical) sensors which are reliable and suitable for motion sensing, (b) real-time sampling of the data and (c) motion detection for a large number of motions (vocabulary), tunable to the needs of different persons. The system is presented in detail in [1].

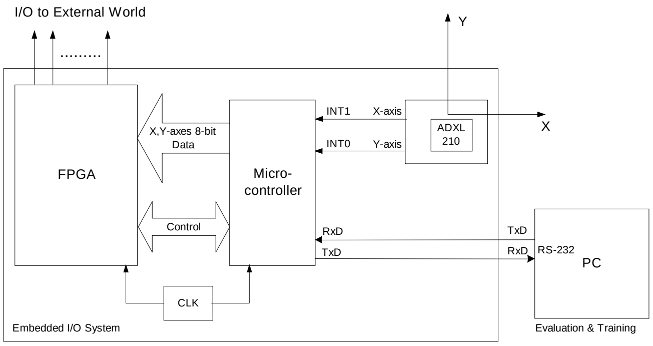


Figure 1. Architecture of the embedded system.

The key decision for the system was the usage of solid-state accelerometers. Although other sensors were considered (e.g. Hall-effect sensors), accelerometers were deemed to be sufficiently small in size, reliable, and low cost. Regarding the computational sub-system, we showed in [1] that a model of independently operating finite state machines (FSM) offers a good design tradeoff vs. the usage of micro-controllers (or microprocessors) alone for free space motion detection. Furthermore, we showed that the sampling and conditioning of real-time data is best performed by micro-controllers, leading to a hardware organization with fixed as well as reconfigurable resources. We also showed that reconfigurable logic is advantageous as compared to VLSI. The VLSI approach lacks the flexibility to solve the general problem at various levels of accuracy vs. number of detected motions at a fixed cost, as indeed is the case for the problem in question. Having the above observations in mind, we established the architecture shown in Figure 1.

A digital two-axis accelerometer (Analog Devices ADXL210) is attached to the user's hand in order to acquire data during hand motions. An 8-bit micro-controller (ATMEL AVR 90S8515) has been used for the sampling of sensor data in real time. The micro-controller undertakes the task of converting the PWM outputs of the sensor to acceleration values. Then, the data are sent to an FPGA (Xilinx SpartanII XC2S100) via an 8-bit bus and to a PC via the serial port. The purpose of the embedded FPGA is to distinguish programmable types of motions. The ability to connect with the PC was employed as a rapid prototyping tool for algorithm evaluation, and as a user interface for system training (to adapt to individual needs), but not as a necessary component during field deployment. *In the standalone operation of the system the personal computer is not connected.* The total system cost is less than $70, an arbitrary limit, which nonetheless precluded certain types of solutions to the motion detection problem. We must notice that the accelerometer is calibrated when it is powered on. This way, the system can be adjusted to different orientations and also to different accelerometers (there are minor differences due to manufacturing variations), when the original sensor has to be replaced. Moreover, the system can operate in different temperature conditions.

4. MOTION REPRESENTATION SCHEME – THE USE OF A DOMINANT AXIS

During preliminary clinical evaluation of the system, many experiments had been performed, and thirteen motions had been successfully sampled and processed [1]. However, it turned out that a large number of complex motions was undesirable to the user, regardless of system capability. For example, circular motions are more complicated than basic motions (such as "left", or "forward") and they require more effort in order to be realized. Therefore, we concluded that a vocabulary comprised of simple motions (forward, back, left, right movement) and their combination is preferable [2].

For the system to be able to recognize these predetermined hand motions, a motion representation scheme is necessary for the description of each movement. The sensor samples hand movement in 2D space. That is, the accelerometer acquires data in two different axes (called X and Y), which are orthogonal. A subset of the collected data in each sampling period corresponds to X axis, and another subset corresponds to Y axis. Therefore, each motion corresponds to a pair of sets. The first set includes sampling data for X axis, whereas the second set includes sampling data for Y Axis.

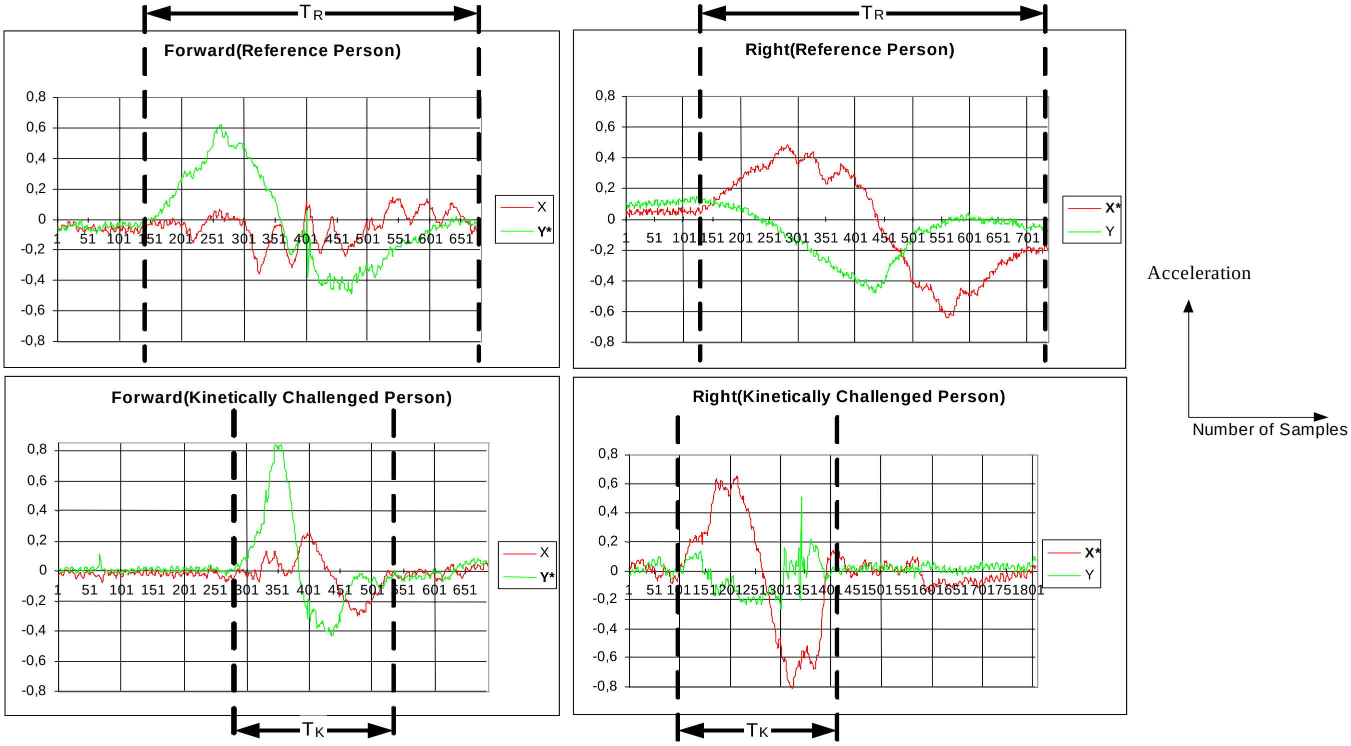


Figure 2. Acceleration charts for motions "forward" (Primary axis: Y) and "right" (Primary Axis: X) (Reference Person vs Kinetically Challenged Person), T_K<T_R.

In terms of processing complexity, the obvious solution is to use a data processing unit per axis. However, experimental results have shown that we can avoid such an approach. As it is shown in Figure 2, one of the two axes (either X or Y) has higher values than the other, for each one of the simple motions. This dominant axis (which corresponds to greater hand mobility) contains a great percentage of the motion's descriptive information. This axis will be called primary, while the other axis will be called secondary.. The latter does not contain stand-alone information, which can differentiate the specific motion from all other motions, due to the fact that the sampling data are characterized by low acceleration values. It can only provide supplementary data, which can be combined with the samples of the primary axis, in order to compose useful information. The above observations stand for the motions of a kinetically challenged person, as well as for the motions of a reference person (with no kinetic problems).

As a result, we are able to describe a specific motion, based on the samples of its primary axis and on the combination of this data with the samples of its secondary axis. This approach decreases the workload of the processing unit, which focuses on the data of the primary axis. By this way, the necessary resources in reconfigurable logic (and the corresponding system cost) are decreased. More specifically, each motion is represented by a set of thresholds that need to be exceeded, for the motion to be recognized. The calculation of the value of these thresholds is based on the data of the primary axis. Regarding the secondary axis, we only need to designate upper and lower bounds for the acceleration. These bounds are predetermined for each motion and set to +0.3g and -0.3g respectively. The reason for such a choice is based on experiments and is presented in Figure 3. As we can see, a hard restriction on the secondary axis of a motion (Figure 3(a)), leads to inability to cope with errors during the execution of a motion. On the contrary, a very loose restriction on the secondary axis (Figure 3(c)) permits too many false positives during the execution of the motion. In this case, overlapping of the motions may take place. In order to avoid both these cases, we determined "middle-of-the-road" bounds for the secondary axis, at 0.3g. This choice ensures that our system can detect motions similar to the one presented in Figure 3(b), i.e. small deviations are acceptable.

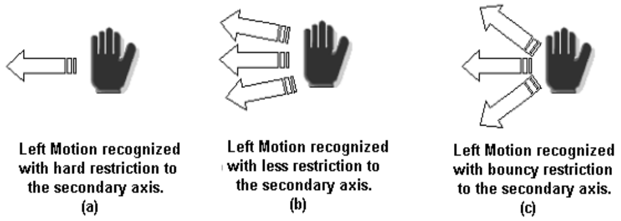

Left Motion recognized with hard restriction to the secondary axis.
(a)
Left Motion recognized with less restriction to the secondary axis.
(b)
Left Motion recognized with bouncy restriction to the secondary axis.
(c)

Figure 3. Examples of a motion with various restrictions to its secondary axis

Focusing on the thresholds of the primary axis, we mainly describe acceleration peaks. Since these peaks are observed in different time spaces, we need to insert a delay between the detection of two consecutive thresholds, in order to avoid local minima (from irregular motion or noise). Therefore, the final motion representation scheme comprises of acceleration thresholds for some axis of interest, each of which is followed by inactive time intervals. The scheme is shown in Figure 4. In this example, the primary axis is Y and the thresholds exceed when an acceleration sample is smaller than -0.4g (1st threshold), another one is greater than –0,4g (2nd threshold), a third one is greater than 0,4g (3rd threshold) and a final acceleration sample is smaller than 0,4g (4th threshold). It is obvious that the scheme is flexible, regardless a motion complexity. More complex motions can be represented by more thresholds and less complex motions are represented by less thresholds.

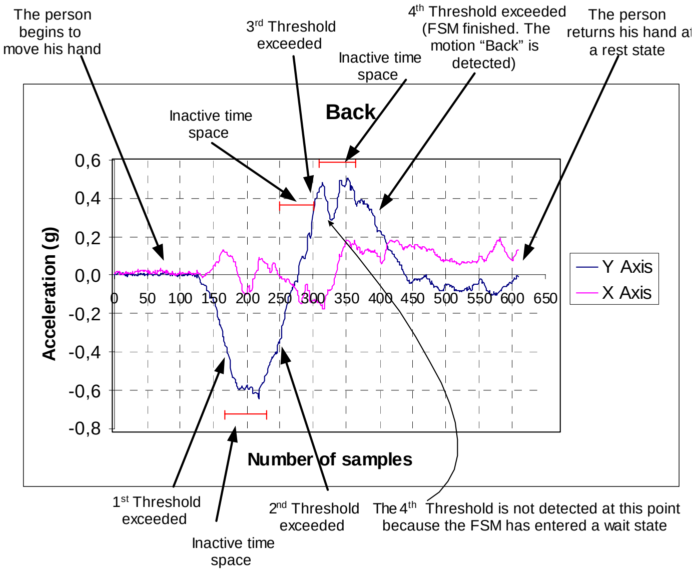


Figure 4. Example of a motion's approximation using thresholds

5. MOTION DETECTION AND SYSTEM ADAPTATION

5.1 Motion Detection Model

The computational model of the system is based on the motion representation scheme and is that of parallel FSMs (Finite State Machines), each of which is comprised of stages for detection of values/ranges of X-Y data, followed by stages to wait for a predefined period of time (including zero time). This way each motion is represented in terms of thresholds, which need to be exceeded for the state to be active, followed by periods of "not examining the input" [1].

As it has already been mentioned, a simple vocabulary of motions that can be used in succession can lead into more options for the user. Therefore, the model in which we concluded, is based on the concept that sequences of the four simple motions (forward, back, left, right movement) are used in order to produce a complex "vocabulary" [2]. This approach allows for a succession of two motions with n possibilities each, to produce n^2 distinct complex motions (n is the number of the FSMs implemented). The FSMs that are integrated in the design are only those that led to the detection of the four simple motions. The complex motions are segmented to two simpler motions and the four FSMs (of the simple motions) are re-used for each segment.

2nd Motion 1st Motion	FORWARD	BACK	LEFT	RIGHT
FORWARD	X	↑ ↓	↑ ←	↑ →
BACK	↓ ↑	X	↓ ←	↓ →
LEFT	← ↑	← ↓	X	← →
RIGHT	→ ↑	→ ↓	→ ←	X

Figure 5. Complex Motion Vocabulary

In order to avoid false positives by successions of the same motion (e.g. “forward – forward”), these cases are not considered as complex motions, leading to n^2 - n well–defined complex motions. *The maximum number of complex motions* that the system can detect is $4^2 – 4 = 12$, which are presented in Figure 5. In addition, it recognizes the simple motions as such (e.g. we detect “forward” but do not look for “forward-forward”), leading to 16 well-defined motions (four simple and twelve complex).

5.2 System Adaptation to Individual needs

It can be said that the general form of each simple motion is the same, regardless the specific user. However, even if this procedure is similar for all users, it is not exactly the same. Clinical tests (of limited scope) have shown that the motions of a kinetically challenged person are not as even as the respective motions of a healthy one. The speed of execution may vary even during the same motion. Thereby, a motion can be too fast in the beginning of the execution and too slow at the end, or vice-versa. The above observations led to the conclusion that the system must have the capability to adapt to the different executions of the same motion by different users.

The idea of adaptation primarily affects the FSMs that are responsible for motion recognition and more specifically the determination of the FSMs’ thresholds that must change according to each user. The procedure that has been implemented for the calculation of the thresholds is an unsupervised learning method. It comprises of two sub-processes: (a) digital representation of the motion’s accelerations graph and (b) calculation of the appropriate thresholds, based on the previous approximation.

5.2.1 Digital representation of a motion

The basic problem that we had to overcome was the local extrema that may appear during the execution of a motion, as it is shown in Figure 6. Due to many factors, like physiological tremble, wrong movement, and deviations from the calibration norm, motions are not even. Local maxima or minima may appear and significantly influence the result. The best way to address this problem is to find the areas of samples that have such a behavior and ignore them. In order to find such samples, we compare each sample with the four previous ones. If the difference between the value of the current sample and the value of its previous ones is greater than a predefined acceleration value, which is considered as the physiological difference between adjacent samples, then the sample belongs to an area of local extrema and is ignored. The above procedure is a kind of bouncy filtering, which offers a good solution to a specific problem and keeps the processing complexity low.

Having in mind the consideration of local extrema, described above, we can go on with the approximation of the accelerations in the graph. As the acceleration samples regard two axes (X, Y), the procedure could be repeated twice. However, according to the motion representation scheme described previously, we need to determine thresholds only for the primary axis of the motion.

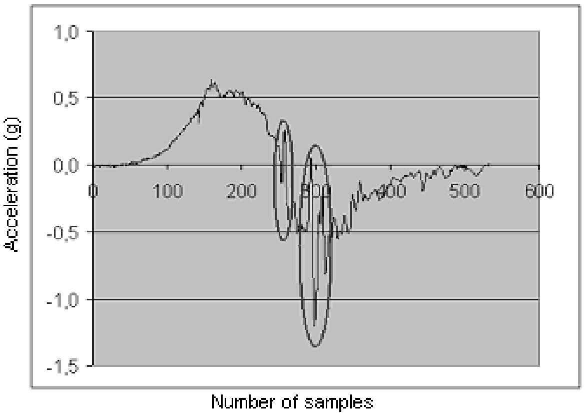


Figure. 6. Local Extrema of a motion

We find the primary axis of the motion, by checking the peak-to-peak distance of the two axes. Then, we approximate the respective acceleration graph. The basic concept is to divide the graph in windows of constant size and transfer the problem to the approximation of these windows. Each of them can be incremental, decremental or stable. We use three counters in order to find the category to which a window belongs, one per category. In each window, we examine the relation of each sample with its previous one and according to the result we increment the respective counter. As explained in the previous paragraph, we ignore samples that belong in areas of local extrema. After examining the last sample of the window, we can determine the category to which it belongs.

The result of the approximation process is a vector, whose positions correspond to the respective windows, as it is shown in Figure 7. The value of each position determines the respective window category (1 for incremental, -1 for decremental, 0 for stable). Based on this vector, we are able to find sequences of windows that lead to threshold calculation.

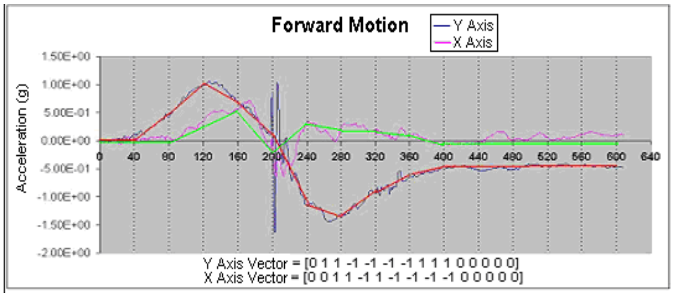


Figure 7. Example of motion approximation

5.2.2 Calculation of a motion's representative thresholds

In general, thresholds are assigned to sequences that correspond to a set of accelerations followed by a set of decelerations, which is the general form of a hand movement. More specifically, the algorithm that finds threshold positions and calculates thresholds is described in Table 1.

Table 1. Sequences of windows that lead to threshold position detection

	Windows				
State	**A**	**B**	**C**	**D**	**Action**
0	1(-1)	-1(1)			Remove A (if inserted). State=3
0	1(-1)	1(-1)			Remove A (if inserted), Insert B. State=0
0	1(-1)	0			State=1
1	1(-1)	0	-1(1)		Insert A, Insert C (if not inserted). State=0
1	1(-1)	0	1(-1)		Remove A (if inserted), Insert C. State=0
1	1(-1)	0	0		State=2
2	1(-1)	0	0	-1	Insert A, Insert D (if not inserted). State=0
3	1(-1)	-1(1)	-1(1)		Insert A, Insert C (if not inserted). State=0
3	1(-1)	-1(1)	0		Insert A, Insert B (if not inserted). State=1
3	1(-1)	-1(1)	1(-1)		State=4
4	1(-1)	-1(1)	1(-1)	0	Insert C (if not inserted). State=0
4	1(-1)	-1(1)	1(-1)	1(-1)	Insert D (if not inserted). State=0

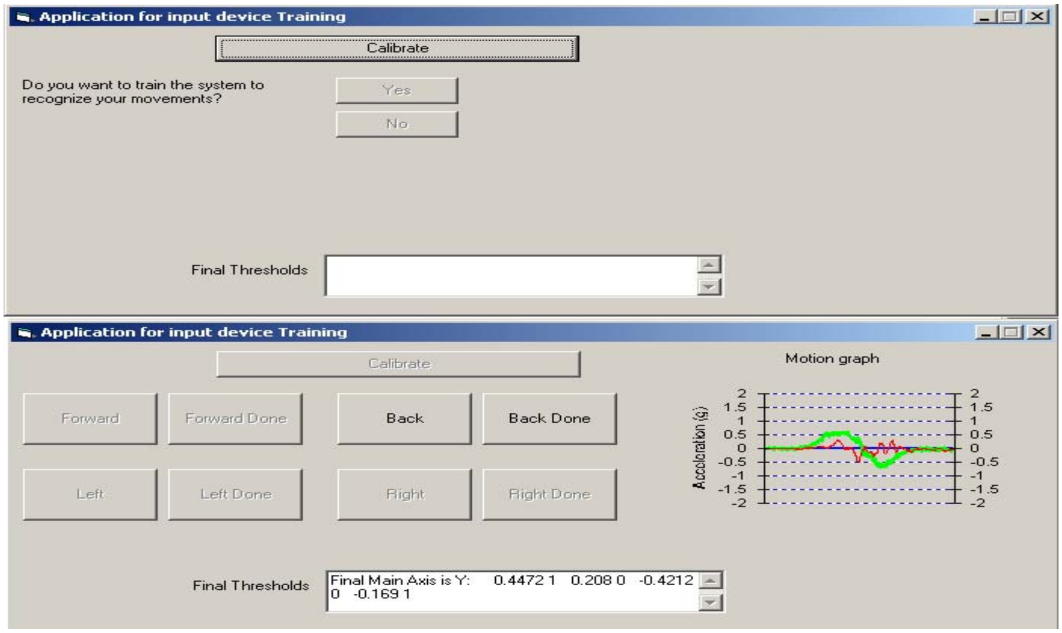
Threshold positions are stored (temporarily or permanently) in a vector. When the scan of all the sequences has been completed, we are able to calculate the thresholds, based on this vector, as follows:

-If a threshold position corresponds to an incremental window then the threshold value is the 80% of the maximum value of the window.

-If a threshold position corresponds to a decremental window then the threshold value is the 80% of the minimum value of the window.

5.2.3 Unsupervised Training Application

By using the PC as the user interface, we collect data for each simple motion from individuals. These data are the patterns, on which the adaptation procedure is based. After calculating the appropriate thresholds for each simple motion following the procedure that was previously described, we download them to a nonvolatile memory, in order to be available for further use. The AVR micro-controller (which is used in the present version of the system) offers an on-chip memory (EEPROM). The micro-controller loads the thresholds from his ROM on power-up and sends them to the FPGA, where they are stored in registers. The FSMs have access to these registers and can read the respective thresholds. In this way, they can recognize the motions that are similar to the initial patterns the user provided. This procedure leads to a system with capabilities to adapt to individual needs but without recompilation of the design. The FPGA is initialized with a fixed design and with changeable input data after it is configured. The training application that undertakes the procedure of obtaining the user's patterns and calculating the appropriate thresholds, is presented in Figure 8.



Do you want to train the system to recognize your movements?

Final Main Axis is Y: 0.4472 1 0.208 0 -0.4212 0 -0.169 1

Figure 8. The Training Application Interface

We notice that in the case that no training has taken place, the operation of FSMs is based on default thresholds. The calculation of these fixed thresholds was based on experimental results. The default operation works very well for healthy persons, however it is likely to encounter problems when the user is kinetically challenged.

6. EXPERIMENTAL RESULTS - CONCLUSIONS

One of the significant results of the experiments was that the preferred motions for the person with kinetic challenges are forward, back, left, right, something that led to the motion detection model. The results of the experiments were valuable in more ways too. The motion representation scheme and detection model works very well for the (generally irregular) motions that a person with kinetic disorders can realize.

The model can recognize simple motions with better than 95% success. Complex motions have 80%-95% recognition rate. Regarding those complex motions whose effectiveness is lower (80%), the change of the initial calibration norm, after the execution of the first segment of the motion contributes to this low percentage, as it is shown in Figure 9.

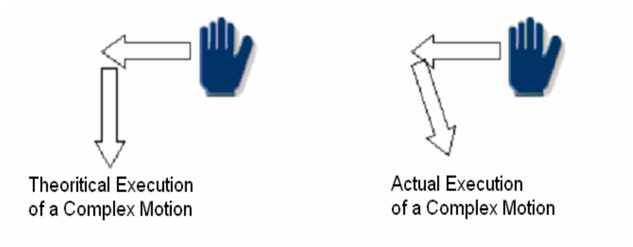


Figure 9.Deviations from the Theoritical Execution of a Complex Motion

Figure 10 presents examples of motion approximation and thresholds' calculation for motions executed by a reference person and a kinetically challenged person. The results are satisfactory in both cases.

Concluding, we have presented a 2D motion detection model, which is readily implementable in reconfigurable hardware for a low-cost solution. The motion representation scheme, in which we concluded, decreases processing complexity. Therefore, the system has diminished demands in reconfigurable resources, something that contributes to the low cost of the whole design. Furthermore, a simple vocabulary of motions can be used to form more complex motions. The model is a good compromise cost/performance-wise, as it can be trained to individual users' needs and it can be extended to more motions, or more complex ones.

7. ACKNOWLEDGMENTS

We wish to acknowledge funding for the initial stages of this work by the Greek Secretariat of Research and Technology (GSRT) and the British Council, under the Britain Greece Joint Research Program, and, the EPET II European Union program under the Second Framework of Support to Greece. We also acknowledge generous donations to our educational and research activities by Xilinx Corporation, and electronics parts support by Analog Devices. We wish to thank Dr. Tom Kean for his contributions and his valuable advice throughout this project. Lastly, we thank Mr. Koutsorinakis for his willingness to test our system in various stages during its development.

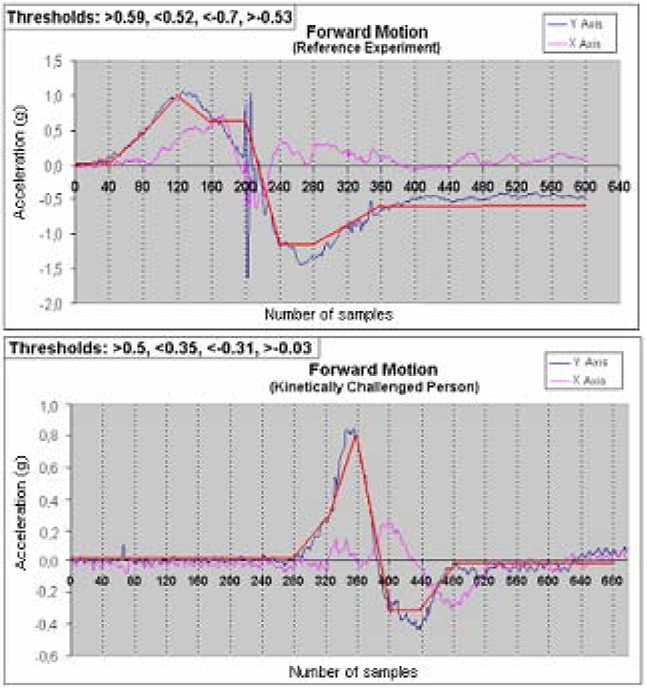


Figure 10. Example of motion approximation and thresholds' computattion

8. REFERENCES

[1] A. Dollas, K. Papademetriou, N. Aslanides, T. Kean, "A Reconfigurable Embedded Input Device for Kinetically Challenged Persons", Proceedings of the European conference on Field Programmable Logic, Belfast FPL2001, 27th-29th of August 2001.

[2] K. Papademetriou, A. Dollas, S. Sotiropoulos, "A Second Generation Embedded Reconfigurable Input Device for Kinetically Challenged Persons", Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines (FCCM), April 2003.

[3] V. Kumar, T. Rahman, V. Krovi, "Assistive Devices for people with Motor Disabilities", Wiley Encyclopedia of Electrical and Electronics Engineering, 1997.

[4] Lau C., O' Leary S. "Comparison of computer input devices for persons with severe physical disabilities", Am. J. Occup. Ther. Nov. 1993, 47:11.

[5] Douglas Hobson, Elaine Trefler, "Rehabiliation Engineering Technologies: Principles of Application", The Biomedical Engineering Handbook - Second Edition, Joseph D. Bronzino, Vol. II, pp. 146-1 – 146-9, 2000.

[6] L. M. Bergasa, M. Mazo, A. Gardel, J.C. García, A. Ortuno, and A.E. Mendez, "Guidance of a Wheelchair for Handicapped People by Head Movements", in Proc. 7th Int. Conference on Emerging Technologies and Factory Automation, ETFA' 99, Barcelona, Oct.1999, pp. 105-111.

[7] R. Barea, L. Boquete, M. Mazo, E. López A. García-Lledó, *"EOG Technique to guide a wheelchair"*, Proceedings of the 16th IMACS World Congress, 2000.

[8] S Mann, C. Aimone, R. Fung, A. Khisti, M. Varia, "Bluetooth Wheelchair", IEEE CSIDC competition, http://www.wearcam.org/wheelchair.htm.

[9] Mary Bond, "Virtual Reality: A significant Tool for Rehabiliation and the Disabled Community", Submitted as partial fulfillment for CS685 001, December 1, 2000.

[10] Laura Dipietro, Angelo Sabatini, Paolo Dario, "Evaluation of an instrumented glove for hand-movement acquisition", *Journal of Rehabilitation Reasearch and Development*, Vol. 40, No. 2, pp. 179-190, March/April 2003.

[11] Lars Bretzner, Ivan Laptev, Tony Lindeberg, Soren Lenman, Yngve Sundblad, "A prorotype System for Computer Vision Based Human Computer Interaction", Technical Report ISRN KTH/NA/P-01/09-SE, April 2001.

[12] Shinji Miyazaki, Akimasa Ishida, Atsushi Komatsuzaki, "A clinically oriented video-based system for quantification of eyelid movements", IEEE Transactions on Biomedical Engineering, Vol. 47, No. 8, August 2000.

[13] Wei Tech Ang, Cameron N. Riviere, Pradeep K. Khosla, "An Active Hand-held Instument for Enhanced Microsurgical Accuracy", Proceedings of the third International Conference on Medical Image Computing and Computer-Assisted Intervention, Pittsburgh, PA, October 11-14, 2000.

[14] Katherine Compton and Scott Hauck, "Reconfigurable Computing: A Survey of Systems and Software", ACM Computing Surveys, Vol. 34, No. 2, pp. 171-210, June 2002.

[15] K. Nakahara, D. L. Jaffe, and E. E. Sabelman, "Development of a Second Generation Wearable Accelerometric Motion Analysis System", 2nd VA National Department of Veterans Affairs Rehabilitation Research and Development, Conference Proceedings "The Next Generation" - February 20 - 22, 2000 - Arlington, VA.

[16] T. Ishimatsu, N. Irie, O. Takami, *"Computer Interface Device for Handicapped People Using Head Movement"*, IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, August 20-22, 1997, Victoria, B.C., Canada.

[17] Ruei-Xi Chen and Liang-Gee Chen, "Design and Implementation of FPGA Wheelchair Controller", Proceedings of the 10th VLSI Design/CAD Symposium, Nantou, Taiwan, Aug. 18-21, 1999, pp57-60.

[18] Manuel Mazo etc, "An Integral System for Assisted Mobility", IEEE Robotics & Automation Magazine, Volume: 8 Issue: 1, March 2001.

[19] Kayser-Threde GMBH, "3D Eye Tracking Device", Technical Description, November 2000, Munchen.