HEAD TRACKING ALGORITHM FOR HUMAN BALANCE EVALUATION

A. D. P.Bregolin*, M. R.Martins*, L.Novelo*, L. P.Prestes**, R. S.Grossi*, A. R.Franco* and D. F. G.Azevedo*

* Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, Brazil
**Uniritter, Porto Alegre, Brazil
email: bregolin@gmail.com

Abstract: This paper describes the development of a new algorithm to measure head movement stimuli in the analysis of human balance. It provides the amplitudes and angular velocities of the selective head movements performed during measurement of vestibular-ocular reflex (VOR). The implementation of the algorithm developed for this study processes three-dimensional data and colored mark detection acquired by the Microsoft sensor Kinect for Windows. The colored marks were affixed to the VOR image acquisition device. Tests were conducted to evaluate the accuracy of the instrument in the measurement of the head movements, using a servomotor device, which mechanically simulates the induced selective head movements.

Keywords: Image processing, vestibular system, movement tracking, Kinect sensor, human balance.

Introduction

The Vestibular System (VS) is responsible for maintaining human balance. Several diseases and conditions affect human balance. Our previous work performs VS evaluation by measuring the response of the system to certain external stimuli: selective head movements that generate compensatory eye movements through the Vestibular-Ocular Reflex (VOR) [1] in order to maintain the line of sight. Images of these movements are acquired by video cameras under IR illumination, digitized by a frame grabber, and digitally processed by a computer. The video cameras are attached to goggles affixed to the patient's head [2].

In this work, we developed a new algorithm to quantify head movements that are performed as stimuli during VS evaluation. The implementation uses digital signal processing of the three-dimensional data acquired by the Microsoft Kinect sensor for Windows. Head movements are quantified by detecting the positions of the color marks attached to the goggles.

We evaluate the VOR response with three different head rotation movements: horizontal (yaw with 30º pitch tilt, to mimic the lateral semicircular canal [3]), vertical (pitch) and torsional (roll) [4]. Our algorithm quantifies these three head movements.

Materials and methods

The main devices and tools used in the development of the project were:
Kinect for Windows Sensor - a line of motion sensing input devices by *Microsoft* [5].
Kinect for Windows SDK - provides the tools and APIs, both native and managed, necessary for the development of Kinect-enabled applications [6].
Computer - implements digital image processing.
TowerPro MG995 - servomotor.
Netduino - microcontroller.
Algorithm - the algorithm was implemented with C# programming language and wich was compiled with Visual Studio 2010 in a windows project. It uses standard image processing techniques, such as color conversion and thresholding, combined with new techniques developed in this work to process three-dimensional data generated by the Kinect sensor. This algorithm tracks the movements of color marks that are attached to the VOR goggles. The HSV (Hue, Saturation and Value) color model are chosen in this work.

The marks are attached to the same plane, affixed on a white board that does not bend during angular movement (Figure 1) and installed on the VOR goggles. The three marks have the same geometric shape, size, and color: 2.2x2.2 cm yellow squares.

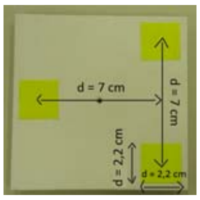


Figure 1: Board with the Fiducial Color Marks.

A block diagram of the developed algorithm is shown in Figure 2. The description of each block is described below:

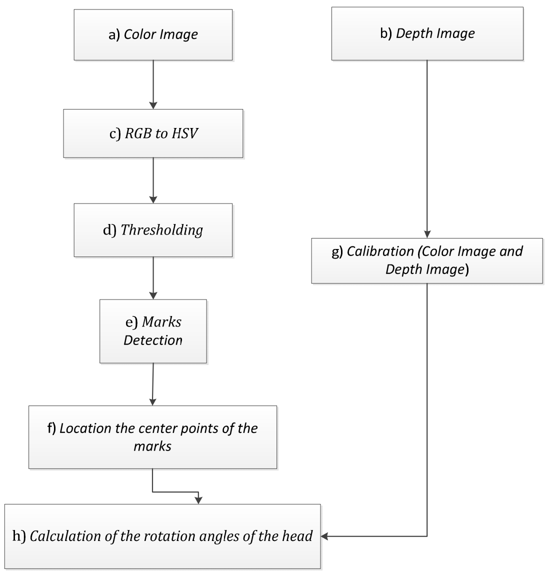

a) Color Image
b) Depth Image
c) RGB to HSV
d) Thresholding
g) Calibration (Color Image and Depth Image)
e) Marks Detection
f) Location the center points of the marks
h) Calculation of the rotation angles of the head

Figure 2: Block Diagram of the developed algorithm.

-**Color Image –** the color image is generated by Kinect sensor in RGB color model.
-**Depth Image –** in parallel to the color image, the image depth of the scene is generated by the Kinect sensor.
-**RGB to HSV – t**he color image is converted from Red, Green, Blue color model (RGB) to Hue, Saturation, Value color model (HSV) in order to optimize the segmentation of the fiducial mark.
-**Thresholding –** the HSV color image is subjected to the thresholding process, resulting in a binary image which is used to detect and segment the fiducial marks.
-**Mark Segmentation –** the fiducial mark detection algorithm extracts the pixels of the Region of Interest (ROI) that are 8-neighborhood connected. That process has a filter to check for minimum and maximum possible sizes.
-**Location of the center points of the marks –** the center of each mark (*Xc, Yc*) is calculated by averaging the *x* and *y* coordinates of all pixels in each mark. The calculation of each center is mathematically expressed by Equation (1).

$$Xc = \frac{1}{N}\sum_{i=1}^{N} Xi\,, \qquad Yc = \frac{1}{N}\sum_{i=1}^{N} Yi \tag{1}$$

-**Calibration (Color Image and Depth image) –**Due to the slight differences in positioning the color camera and the depth sensor, there is an error caused by parallax to determine the depth of each pixel in the color image. This error is corrected by software calibration using the *Kinect for Windows SDK*.
-**Calculation of the rotation angles of the head –** The method to calculate the rotation angles of the head *yaw* and *pitch*¸ is divided into two stages. In the first stage, it is necessary to calibrate the reference point for each head movement: the initial angle of motion (0 °) is obtained when the center points of the fiducial marks are at the same depth, and considering that they are aligned in the same plane (Figure 3). At this position we determine the number of pixels in the color image corresponding to the distance between the marks. This value is designated *d₀*.

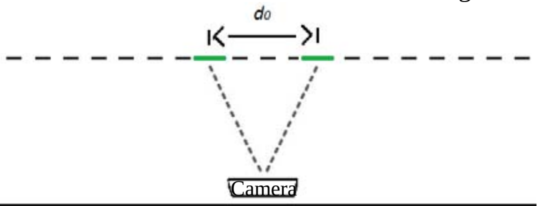


Figure 3: Initial Calibration of the Algorithm.

In the second stage, the head movements are monitored by tracking the mark positions in the projected plane of the camera, in order to measure the actual distance (in pixels) between them. This value is called d_0'. By projecting the distances d_0 and d_0' in three-dimensional space, a rectangle triangle is generated, in which the hypotenuse is d_0 and d_0' is the adjacent side (Figure 4).

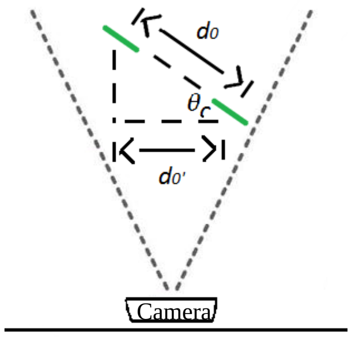


Figure 4: Triangle Rectangle Project in the three-dimensional in the projected plan of the camera.

The angle θ between the distances determines the angular movement of the head. Which can be calculated by Equation (2).

$$\theta = \cos^{-1}\left(\frac{d_0'}{d_0}\right) \tag{2}$$

To determine the rotation angle roll (torsional plan), the slope of the line formed between the center of the fiducial marks is calculated by Equation (3).

$$\theta = \tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \tag{3}$$

Results

The C# programming language was used to implement the algorithm proposed by this work. The core interface of the application (Figure 5) tracks the amplitude of the head movements *yaw*, *pitch* and *roll*. This interface imports the data from the configuration files created by previous stages (thresholding and calibration) to perform the calculations. The core interface exports the quantification charts to be later interpreted by an investigator.

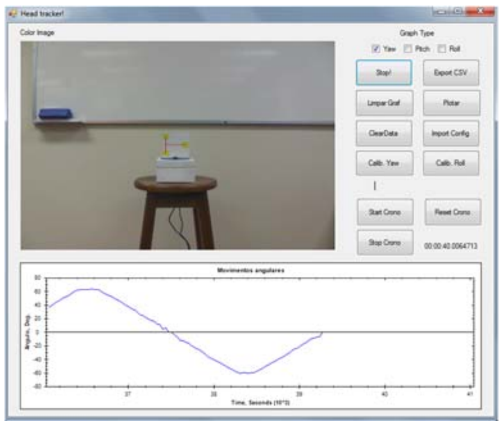


Figure 5: Core Interface Developed to the Algorithm.

In order to evaluate the accuracy of the instrument in quantifying the stimulus (selective head movements), we developed a device that simulates the head movements executed during the VOR quantification (Figure 6). This device simulates the head angular movements *yaw*, *pitch* and *roll*, and generate stimuli with controlled amplitude and angular velocity. This device uses the netduino plus 2 board and the towerpro MG995 servomotor.

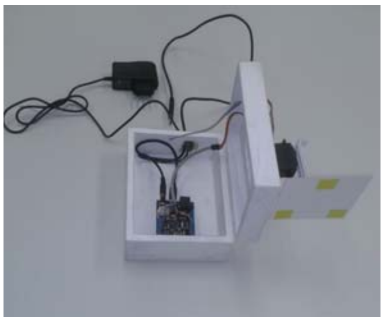


Figure 6: Device developed for the tests.

This test was developed with 20 repetitions for each tested amplitude, for each of the three intended angular movements. The test for measuring angular velocity of the stimulus was conducted to define the precision of the measurement of this parameter. This test was also performed using the simulator of angular movements with 20 repetitions.

The quantified angular movements of this test (*yaw*, *pitch* and *roll*) was displayed in graphs with the axes in degrees [°] and time in seconds [s].

Below, we show the description of all the tests and their results:

-**Accuracy Test:** This test aims to determine the accuracy of the algorithm in quantifying the stimulus using the yaw angular movements. The test was performed with sinusoidal movements originating from the reference angle (0°) at the different amplitude values: 61°, -60°, 35°, -34°.

In the graph of the *yaw* angular movement, the positive semicircle represents movement to the left, and the negative semicircle, movement to the right. The graph corresponding to this movement can be seen in Figure 7.

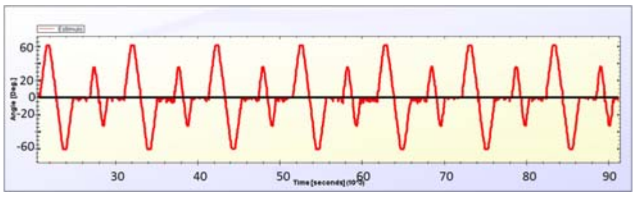


Figure 7: Graph corresponding to the *yaw* angular movements. X axis is the time and Y axis the angle

The average and the standard deviation values of each detected peak in this achieved quantified movement were:
-**Amplitude 1**: 60,948735656°+- 0,486728124601595°
-**Amplitude 2**: -60,579996814°+-0,366383077364153°
-**Amplitude 3**: 35,209773856°+-0,861015158432058°
-**Amplitude 4**: -33,908738899°+- 1,18831321900246°

-**Accuracy test (pitch angular movements)**

This test aims to determine the accuracy of the instrument in quantifying the stimuli using the *pitch* angular movements. The test was performed with sinusoidal movements originated in the reference angle (0°) at different stimulus amplitudes:
-**Amplitude 1:** -35°;
-**Amplitude 2:** 31°;
-**Amplitude 3:** -35°;
-**Amplitude 4:** -59°.

The graph corresponding to the movement can be seen in Figure 8.

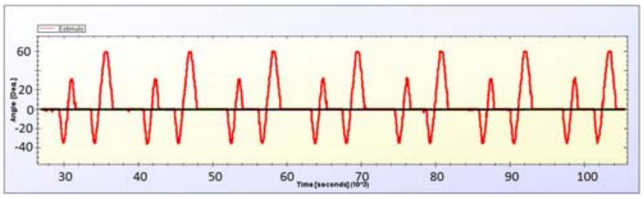


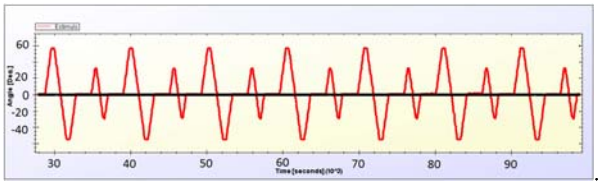
Figure 8: Graph corresponding to the *pitch* angular movements. X axis is the time and Y axis the angle

The average and the standard deviation values of each detected peak in this achieved quantified movement were:
-**Amplitude 1**: -35,46371208°+-1,31681472060246°
-**Amplitude 2**: 31,37517509°+-1,83798625234841°
-**Amplitude 3**: -35,21511237°+-1,37207433993998°
-**Amplitude 4**: 59,14614928°+-0,915794094014368°

-**Accuracy test (roll angular movements)**

This test measures the accuracy of the instrument in quantifying the stimulus using the *pitch* angular movements. The test was performed with sinusoidal movements starting in the reference angle (0°) at different stimulus amplitudes: 57°, -55°, 32°, -30°.

The graph corresponding to the movement can be seen in Figure 9.



.

Figure 9: Graph corresponding to the *roll* angular movements. X axis is the time and Y axis the angle.

The average and the standard deviation values of each detected peak in this achieved quantified movement were:
-**Amplitude 1**: 56,69517101°+-0,210002276550631°
-**Amplitude 2**: -55,37003362°+-0,190024204331485°
-**Amplitude 3**: 31,69287804°+-0,180267123650248°
-**Amplitude 4**: -29,59509510°+-0,437680760992015°

-**Accuracy test (yaw angular movements with a tilt pitch of 30°)**

This test measures the accuracy of the instrument in quantifying the stimuli using the *yaw* angular movements with a *pitch* tilt of 30°. The test was performed with the same sinusoidal movements used in the test *(1)* with 20 replications for each range tested. The graph corresponding to the movement can be seen in Figure 10.

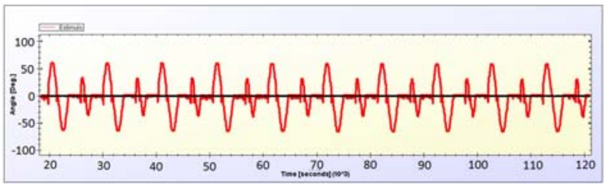


Figure 10: Graph corresponding to the *yaw* angular movements with a tilt *pitch* of 30°. X axis is the time and Y axis the angle

The average and the standard deviation values of each detected peak in this achieved quantified movement were:
-Amplitude 1: 60,88750706° +- 1,80233619961799°
-**Amplitude 2**: -60,20691735° +- 1,36900521120062°
-**Amplitude 3**: 35,10952252° +- 1,78394943564312°
-**Amplitude 4**: -33,82886346° +- 1,31266660191077°

-Accuracy test (angular velocity estimation)

This test was performed using the device that simulates the head movements, and for each test the device performed at constant speed. We used timers in the servomotor control code to calculate the average angular velocities between two consecutive peaks, amounting to 59,08°/s (vel. 1) and 84,03°/s (vel. 2) (degrees/second). Sinusoidal *yaw* motions were used in this test, with 20 repetitions, and used the same amplitudes as in *test (1)*. They produced two graphs with the angular velocities shown in Figure 11 and 12.

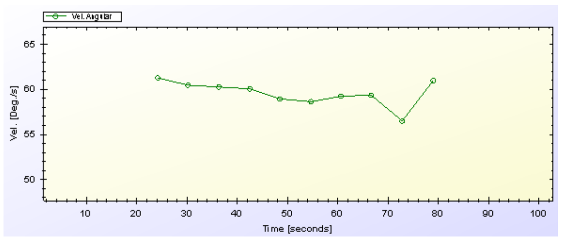


Figure11: Graph corresponding to the angular velocity average for every two peaks detected (vel. 1).

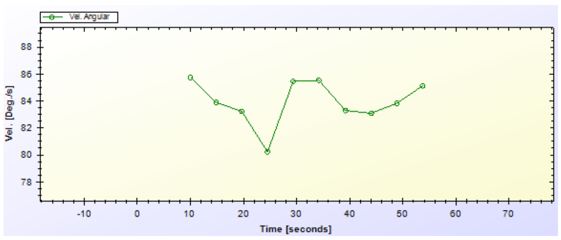


Figure 12: Graph corresponding to the angular velocity average for every two peaks detected (vel. 2).

The average and standard deviations for each average angular velocity, for every consecutive peak interval were measured:
-**Vel. 1**: 59,50959670°/s+-4,17158938961367°/s.
-**Vel. 2**: 83,91614011°/s+-4,99984942101111°/s.

Discussion

The developed application implements the designed algorithm to quantify the stimulus and display, export and import data for a future use by an investigator.

The tests using the simulator device for angular movements, developed in this work, demonstrate the accuracy of the algorithm in the measurement of the amplitude angles and angular velocities.

The results of the parameter extraction of angular velocity performed in this test can be improved through the use of a sensor with faster video rates (FPS) than the Kinect sensor (with 30 FPS), and using a more accurate motor with optical sensors to monitor effective movement of the motor (encoder).

Conclusion

Through the images obtained by *Kinect Sensor,* the software described in this paper is capable of quantifying the movements generated by the head movements emulator.

Acknowledgements

We would like to thank Prof. Azevedo and PUCRS.

References

[1] Mezzalira R, Bitar RSM, Albertino S. Otoneurologia Clinica. 1ºed. Rio de Janeiro: Revinter; 2014.
[2] Figueira MFV, Azevedo DFG, Russomano T, Zaffari CA, Rocha MF. Improvements on a Fast Algorithm for Real Time Eye Movement Quantification [dissertation]. Porto Alegre: Pontifícia Universidade Católica do Rio Grande do Sul; 2007.
[3] Herdman SJ; Vestibular Rehabilitation. Philadelphia: F. A. Davis Company;2007.
[4] Leigh JR, Zee DS. The Neurology of Eye Movements. New York: Oxford University Press; 1999.
[5] Miles R. Start Here! Learn the Kinect API, California: Microsoft Press; 2012.
[6] Webb J, James A. Beginning Kinect Programming with the Microsoft Kinect SDK: Apress; 2012.